VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zpracování fotografie v rámci systému FOTOM NG

Photograph Processing within the FOTOM NG System

2019 Andrej Gulčík

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Andrej Gulčík**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Zpracování fotografie v rámci systému FOTOM NG
Photograph Processing within the FOTOM NG System

Jazyk vypracování: čeština

Zásady pro vypracování:

Cílem diplomové práce je návrh a realizace modulu na segmentaci pór ve vnitřní struktuře vzorků různých materiálů určujících jejich mechanické vlastnosti s použitím systémů Micro-CT a FOTOM. Součástí práce bude také vyhodnocení základních fyzikálních vlastností vzorků ve vztahu k technologii výroby.

1. Seznamte se s problematikou počítačového zpracování fotografií a metod segmentací pór ve vnitřní struktuře vzorků různých materiálů.
2. Seznamte se s prostředím NetBeans a programovacím jazykem JAVA.
3. Seznamte s fotogrammetrickým systémem FOTOM verze NG.
4. Navrhněte nový modul na segmentaci objektů-pór vzorků v rámci systému FOTOM-NG.
5. Implementujte navržený modul v systému FOTOM-NG a proveďte zhodnocení dosažených výsledků.
6. Vypracujte uživatelskou a programátorskou dokumentaci.

Seznam doporučené odborné literatury:

[1] GONZALEZ, Rafael C. a Richard Eugene WOODS. *Digital image processing.* 3rd ed. Upper Saddle River: Pearson Prentice Hall, c2008. ISBN 978-0-13-168728-8..
[2] JAN, Jiří. *Medical image processing, reconstruction and restoration: concepts and methods.* Boca Raton: Taylor & Francis, 2006. ISBN 0-8247-5849-8.
[3] LIANG, Daniel. *Introduction to Java Programminig, Comprehensive.* Upper Saddle River, New Jersey: Pearson Education, Inc., 2015. ISBN 13 978-0-13-376131-3.
[4] LIČEV Lačezar. *Systém FOTOM-NG, architektura, funkce a použití.* Praha: BEN - technická literatura, 2015. ISBN 978-80-7300-521-4.
[5] SOJKA, Eduard. *Digitální zpracování a analýza obrazů.* Ostrava: VŠB - Technická univerzita Ostrava, 2000. ISBN 80-7078-746-5.
[6] BANHART, J.: Manufacture, characterization and application of cellular metals and metals foams. Progress in Materials Science, 46, 2001, pp. 539 - 632

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **prof. Ing. Lačezar Ličev, CSc., prof.h.c.**

Datum zadání: 01.09.2018

Datum odevzdání: 30.04.2019

doc. Ing. Jan Platoš, Ph.D.
vedoucí katedry

prof. Ing. Pavel Brandštetter, CSc.
děkan fakulty

Prehlasujem, že som túto diplomovú prácu vypracoval samostatne. Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

V Ostrave 30. 4 2019 ..... Gulaťk

Rád by som sa poďakoval pánovi prof. Ing. Lačezarovi Ličevovi, CSc. prof.hc za príležitosť, trpezlivosť a odborné vedenie, pánovi Ing. Kamilovi Součkovi, Ph.D. za ochotu a pomoc a v neposlednom rade pani Ing. Ivane Kroupove, Ph.D za trpezlivosť a ochotu. Najväčšia vďaka patrí mojej snúbenici Monike, bez ktorej by táto práca nevznikla. Za podporu, zázemie a trpezlivosť, ktorú somnou mala jej patrí najväčšia vďaka.

Abstrakt

Cieľom tejto diplomovej práce bolo vytvorenie programu, schopného segmentovať póry, pre možnosti analýzy štrukturálnych vlastností kovových materiálov. Preto bolo nutné zoznámiť sa s najpoužívanejšími segmentačnými metódami. V úvode práce sú predstavené dôvody vzniku tejto práce, ako aj postup výroby kovových pien. Následne je uvedené súčasné smerovanie segmentácie vo svete. V ďalších kapitolách sa venujem formátu DICOM a predspracovaniu získaných dát. V ďalšej kapitole sú predstavené rôzne segmentačné metódy, ktoré boli implementované v jazyku Java a zahrnuté do systému FOTOM NG. V závere je predstavená samotná implementácia a bližšie vysvetlené postupy, ktoré boli v práci použité. Záver práce je venovaný zhrnutiu dosiahnutých výsledkov. Výsledkom tejto práce je modul systému FOTOM NG pre segmentáciu kovov v obrazový dátach formátu DICOM implementovaný v jazyku Java. K tomuto modulu bola vytvorená aj programátorská dokumentácia a uživatelská príručka.

Kľúčové slová: Java, segmentácia, FOTOM NG, Prahovanie, Filtrácia, Watershed, kMeans, DICOM, Hrana, Kovová pena

Abstract

The aim of this diploma thesis is a software able to recognize metal cracks to analyze structural properties of metal metarials. It was necessary to understand the most used segmentations methods. In the introductuion of the work is explanation for this thesis and also the methods of metal foams production. Next articel is focused on actual metods of segmentation in the world. Later is focus on DICOM format and pre-processing of obtained data. Next articel introduce different segmentations metods implemented in Java and implicit in to systém FOTOM NG.The end of the diploma thesis contains concret implementation, explanations of mechanisms used in this work and the resume of the results. The result of this thesis is a modul for FOTOM NG system for metal segmentations in DICOM format implemented in Java. To this modul was also created programmer documentation and user manual.

Key Words: Java, segmentation, FOTOM NG, Thresholding, Filtration, Watershed, kMeans, DICOM, Edge, Metal foam

Obsah

Zoznam použitých skratiek a symbolov

FEI	–	Fakulta elektrotechniky a informatiky
FMMI	–	Fakulta materiálovo-technologická
VR	–	Value representation
VL	–	Value length
VM	–	Value multiplicity
S	–	Sever
SV	–	Severo-východ
V	–	Východ
JV	–	Juho-východ
LoG	–	Laplacian of Gaussian
DoG	–	Difference of Gaussians

Zoznam obrázkov

Zoznam tabuliek

1 Úvod

Segmentácia obrazu je jednou z najdôležitejších úloh automatického spracovania obrazu. V podstate je jej úlohou extrahovať z obrazu informáciu o tom, kde sa nejaký objekt prípadne objekty nachádzajú. V prípade umelo vytvorených obrazov nemusí byť táto úloha problémom. Nanešťastie sa v reálnych podmienkach stretávame s mnohými problémami, ktoré kvalitu segmentácie výrazne zhoršujú. Či už sa jedná o nedokonalosť zariadenia, ktorým obraz vyhotovujeme, nerovnomerné prípadne zlé svetelné podmienky až po nejednoznačnosť hranice objektu.

Cieľom tejto práce bolo vytvoriť modul na segmentáciu pór vo vnútornej štruktúre vzorov. Výstup tejto segmentácie bude ďalej použitý na analýzu štrukturálnych a mechanických vlastností týchto vzorov. Tento modul bude implementovaný do systému FOTOM NG. Táto práca vznikla vrámci spolupráce fakulty FEI s fakultou FMMI. Nakoľko existuje viacero zlievarenských postupov, kde každý sa vyznačuje inou časovou, technologickou alebo finančnou náročnosťou ako aj výsledkami, je vhodné mať možnosť ako čo najefektívnejšie porovnať tieto zlievarenské postupy.

Ako už bolo spomenuté výstupom tejto práce je segmentovaný obraz. Tento obraz je neskôr použitý na vytvorenie 3D modelu vnútornej štruktúry vzorku. V tomto module bude najmä možné podrobnejšie analyzovať vzorky, ktoré vznikli rôznymi zlievarenskými metódami a vybrať najvhodnejšiu.

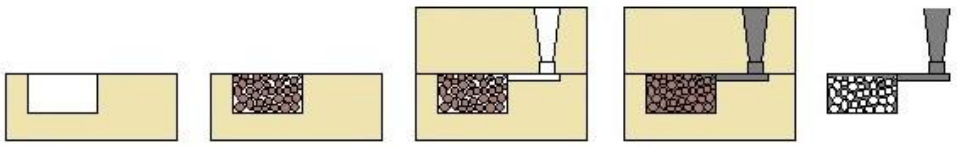
Na začiatku práce sa venujem formátu DICOM, nakoľko všetky testovacie snímky boli v tomto formáte. Ďalšia kapitola je venovaná metódam predspracovania obrazu, za účelom lepšieho výsledku segmentácie. Táto časť pojednáva o základných operáciach s histogramom a predstavuje filtračné metódy použité v tejto práci. Následná kapitola pojednáva o implementovaných metódach segmentácie. Všetky tieto metódy sú taktiež súčasťou implementácie. O implementácii samotnej pojednáva nasledujúca kapitola. Tu sú zároveň vysvetlené prípadne modifikácie algoritmov a zobazená základná štruktúra modulu. Azda najpodstatnejšou časťou je postup, ktorým je z mapy hrán získaný objekt. Predposledná kapitola sa venuje testovaniu metód a ich výsledkom. V tejto časti je predstavený postup testovania ako aj použitý optimalizačný algoritmus. Obsahuje informácie o dosiahnutých výsledkoch. Práca je zakončená záverom, v ktorom sú zhrnuté výsledky testovania, ako aj návrhy budúcej práce.

1.1 Kovová pena

Kovové peny predstavujú materiál so širokým uplatnením v mnohých oboroch ako automobilový priemysel, stavebný priemysel, medicína a podobne. Názov pena vychádza z faktu, že tento materiál obsahuhe vo svojej štruktúre umelo vytvorené póry. Tieto póry im poskytujú vlastnosti ako napríklad vysoká tuhosť pri nízkej teplote, vysoká tepelná vodivosť, schopnosť absorbcie energie a iné [1].

Vnútorné póry kovovej peny môžu byť vytovrené pomocou použitia výplňového materiálu. Tento materiál môže byť prekurzor alebo preforma. Výplňový materiál je vložený do dutiny formy a zaliaty tekutým kovom.Tieto materiály musia spĺňať určité kritéria, ako napríklad musia byť vytvorené z materiálu, ktorý zachová svoj tvar pri styku s tekutým kovom a taktiež musí vykazovať dobrú rozpadavosť po odliatí.

Nepravidelnosť rozloženia vnútorných dutín môžeme dosiahnuť použitím prekurzov voľne nasypaných do dutiny formy. Ak chceme dosiahnuť pravidelnú štruktúru použijeme jadro (preformu) s požadovaným tvarom [1]. Postup zalievania je zobrazený na obrázku 1.



Obr. 1: Princíp infitrácie tekutého kovu do dutiny formy s prekuzormi.

Na obrázku 2a je zobrazený prekurzor prvej generácie, ktorý je tvorený najmä kremíkovým pieskom, ílom a vodou, vložený do formy, ktorá je na obrázku 2b. Výsledný odliatok je na obrázku 2c.

(a) Forma s prekurzorom. (b) Neopracovaný odliatok. (c) Výsledný odliatok.

Obr. 2: Postup odlievania.

Pri analýze kovovej peny by sme chceli sledovať a hodnotiť veľkosť distribúcií jednotlivých zámerne vytvorených pór. Zároveň nás zaujímajú póry, ktoré nie sú žiadané a vznikli v dôsledku fyzikálno-chemických procesov, ku ktorým dochádza v rámci konkrétnej technológie výroby. Taktiež je vhodné analyzovať objemové zmeny, ktoré vznikli pri tuhnutí odliatku, prípadne plynové vady. Analýzou týchto vlastností je možné optimalizovať proces výroby kovovej peny.

2 State of the Art

V súčasnej dobe sa v mnohých oblastich spracovania dát vo veľkom využívajú neurónové siete. Ani segmentácia obrazu nie je výnimkou. Výhodou neurónových sieti je ich robustnosť a parametrizovatelnosť. Taktiež ich implementácia často nebýva zložitá. Napríklad v práci [2] využívajú hlboké učenie a konvolučné neurónové siete na segmentáciu. Vo všeobecnosti sa dnes najmä konvolučné siete tešia veľkej obľube a preto je možné nájsť množstvo prác, ktoré využívajú práve túto metódu.

V práci [3] sa taktiež spomína využívanie neurónových sieti ale aj soft computingu ako takého. Soft computing v sebe zahŕňa okrem iného fuzzy logiku alebo genetické algoritmy.

Práve využitie fuzzy logiky môže viesť k zlepšeniu výsledkov segmentácie pomocou zhlukovacích algoritmov. Tie sú spomenuté napríklad v práci [4]. V nej sa upozorňuje na fakt, že zhlukovacie techniky sú jedny z najpoužívanejších metód segmentácie. Problémom však je, že tieto metódy vyžadujú počiatočné nastavenie uživateľa. Práve správnosť voľby počiatočných parametrov je kritická, aby sme boli schopní získať dobré výsledky. V tejto práci je predstavená nová segmentačná metóda založená na zhlukovaní a odhade hustoty pixelov, ktorá toto počiatočné nastavenie nevyžaduje.

V práci [5] je pre zmenu použitý modifikovaný snake algoritmus na segmentáciu tumorov. Snake algoritmus patrí medzi algoritmy, ktoré využívajú aktívne kontury. Ich nevýhodou je, že vyžadujú inicializačné body, v ktorých algoritmus začne a navyše sú pomerne časovo náročne. Pre úplnosť len dodám, že aj v tejto práci je použitá neurónova sieť, avšak na klasifikáciu samotnú.

Rád by som ešte spomenul súčasné riešenie tejto problematiky v systéme FOTOM NG, keďže výsledkom tejto práce je modul do tohoto systému. Aktuálne obsahuje systém riešenie pre detekciu pór. Ako segmentačná metóda v tomto moduly bolo použité binárne prahovanie.

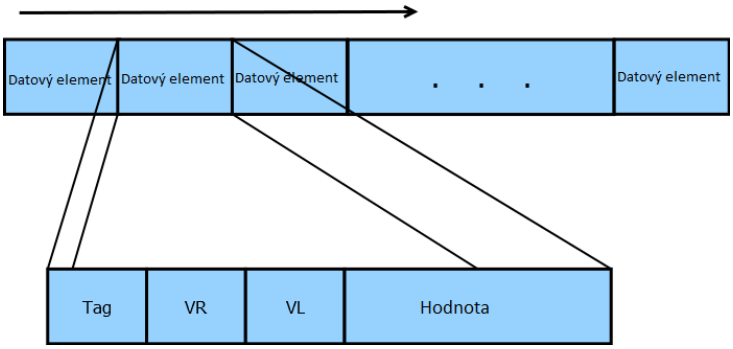
3 DICOM

Štandard DICOM(Digital Imaging and Comunications in Medicine) bol vytvorený v roku 1984 za účelom zobrazovania obrazových medicínskych dat. Okrem samotnej informácie o obraze obsahuje tento formát aj ďalšie doplňujúce informácie, ktoré súvisia so zachytením obrazu.

Všetky dáta použité na testovanie v tejto práci boli dodané práve v tomto formáte. Z tohoto dôvodu uvediem iba základnú štruktúru DICOM súboru, keďže bol pre potreby práce vytvorený aj modul na získanie obrazových dat z DICOMu.

Datova štruktúra začína hlavičkou. Tá obsahuje 128 bajtovú *preambulu*, za ktorou nasledujú 4 bajty obsahujúce znaky D, I, C, M.

Za hlavičkou nasleduje *Dataset*, ktorý sa skladá z *datových elementov*. Jeden DICOM súbor môže obsahovať viac ako jeden *dataset*. Štruktúra datovej časti je zobrazená na obrázku 3.



Obr. 3: Štruktúra DICOM datasetu.

Tag sa skladá z dvoch 16–bitových celých čísel. Tieto čísla reprezentujú skupinu a element skupiny.

VR (Value repesentation) predstavuje dvojbajtový reťazec, ktorý je tvorený dvoma znakmi. Tie reprezentujú datový typ hodnoty.

VL (Value length) predstavuje bajtovú reprezentáciu dĺžky hodnoty.

Hodnota Okrem VR a VL ovplyvňuje hodnotu ešte parameter **VM** (Value multiplicity). Ten vyjadruje koľko hodnôt môže byť obsiahnutých v hodnote elementu.

V dokumente teda vyhľadávam elementy a na základe tagov určím akú informáciu nesú. Túto informáciu získam z hodnoty. Naprílad datový element **(0010,0040) CS 0002 M** obsahuje informáciu o pohlaví pacienta. Skupina 0010 obsahuje informácie o pacientovi, element 0040 predstavuje pohlavie. CS znamená, že sa jedná o reťazec a 0002 informuje o počte bajtov, ktoré je nutné načítať a teda 2. M identifikuje pohlavie samotné.

4 Predspracovanie obrazu

Cieľom predspacovania je prevedenie obrazu do takej podoby, ktorá je vhodná pre následnú segmentáciu. To môže zahŕňať operácie ako prevod na šedotónový obraz, operácie s histogramom, úpravu jasu, kontrastu prípadne filtráciu pre odstránenie šumu v obraze.

4.1 Prevod na Šedotónový obraz

Metódy v tejto práci sú implementované pre šedotónový obraz. Takýto obraz sa vyznačuje tým, že obsahuje iba hodnoty od 0 po 255, kde 0 predstavuje čiernu farbu a 255 farbu bielu. Pri prevode RGB obrazu na šedotónový sú hodnoty prevedené pomocou vzorca 1.

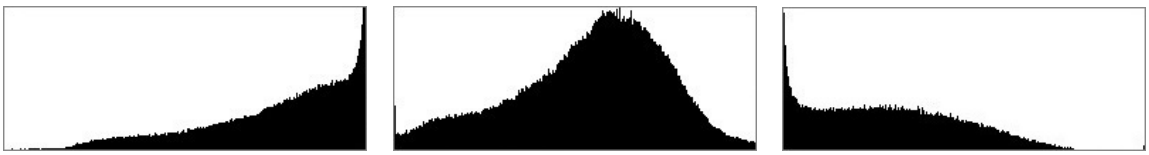
$$I = 0.33R + 0.33G + 0.3B \tag{1}$$

4.2 Histogram

Histogram predstavuje grafické znázornenie početnosti pixelov s určitou hodnotou jasu. Osa y predstavuje úrovne jasu. Pri šedotónových obrazoch nadobúda hodnoty 0 až 255. Pri farebných obrazoch vytváram histogram pre každú zložku zvlášť. Informácie, ktoré z histogramu získam je možné využiť pre vylepšenia vzhľadu - detekovanie preexponovanosti prípadne podexponovanosti. Príklady histogramov sú zobrazené na na obrázku 4. Pomocou vzorca **??** je možné získať normalizovaný histogram. V takom prípade reprezentuje osa y pravdepodobnosť výskytu jednotlivých úrovni jasu.

$$P_{(x_i)} = \frac{n_i}{n}, i \in 0, 1, 2, \ldots, L-1 \tag{2}$$

kde n_i je počet pixelov s intenzitou i a n je počet všetkých pixelov.



(a) Histogram preexponovaného obrázku.

(b) Ideálny histogram.

(c) Histogram podexponovaného obrázku.

Obr. 4: Príklady histogramov.

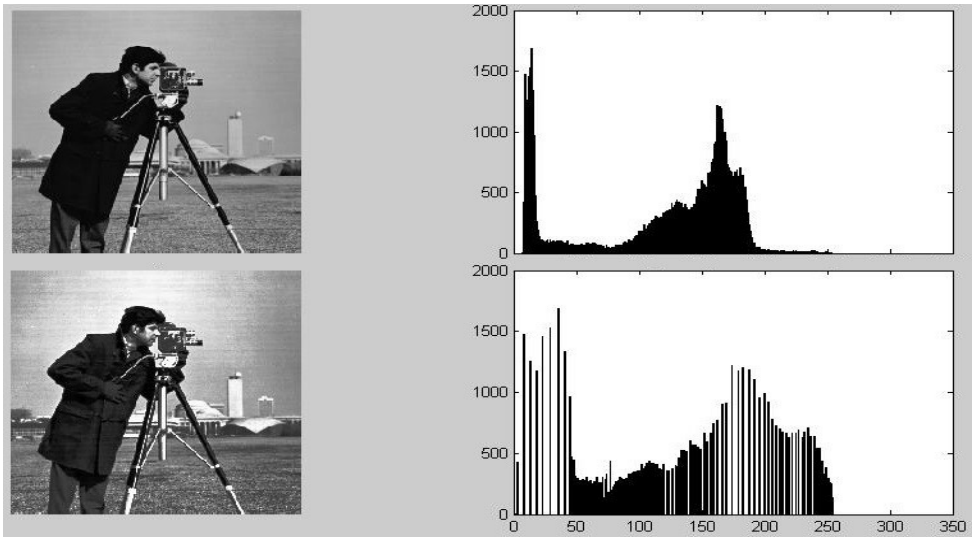
4.2.1 Histogramova ekvalizácia

V prípade, že je obraz preexponovaný alebo podexponovaný je možné vylepšiť obraz použitím histogramovej ekvalizácie. Táto metóda pomôže histogram „vyrovnať“. Snaží sa teda rozložiť početnosť pixelov do celého spektra, keďže preexponované alebo podexponované obrazy majú väčšinu pixelov sústredených iba v určitej časti spektra. Príklad obrazu pred a po histogramovej ekvalizácií aj s príslušnými histogrammi je možné vidieť na obrázku 5.

Pri ekvalizácií je používaná kumulatívna distribučná funkcia, ktorá je definovaná vzorcom 3.

$$c(i) = \sum_{j=0}^{i} P(j) \tag{3}$$

Táto metóda pomáha pri zlepšení kontrastu obrazu, avšak pri segmentácii môže viešt k zhoršeniu výsledkov, keďže môže zvýrazniť pixely pozadia.



Obr. 5: Histogramová ekvalizácia.

4.3 Filtrácie obrazu zahrňujúce okolie

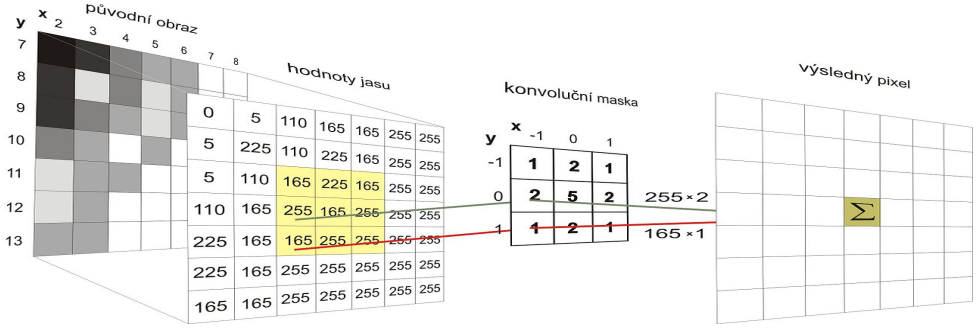
Filtrácie tohoto typu pracujú s okolím bodu, ktorý je spracovávaný. Výsledná hodnota vychádza z hodnôt okolitých pixelov (napríklad priemerná hodnota).

Pomocou týchto fitrov sa snažím odstrániť šum z okolia a tým zlepšiť výsledky segmentácie. Problémom je, že aplikáciou fitrov sa redukuje informácia v obraze, napríklad tým, že sa zjemňujú prechody hrán. Pri silnej filtácii by sa teda stratila informácia o malých objektoch v obraze. Medzi základné vyhladzovacie filtre patria podľa [6] : priemerový filter, mediánový filter a Gaussov filter.

4.3.1 Konvolúcia

Konvolúcia je operácia, ktorá je často využívaná pri spracovaní obrazu. Využíva konvolučné jadro (*kernel*), ktoré sa prekryje s obrazom. Hodnoty na odpovedajúcich pozíciach sa vynásobia a ich súčet sa dosadí na pozíciu skúmaného pixelu. Takto sú získané výsledné hodnoty pre celý obraz. Princíp 2D diskrétnej konvolúcie je na obrázku 6. Pre obraz f a jadro h je definovaná vzorcom 4.

$$(f * h)(x, y) = \sum_{i=-k}^{k} \sum_{j=-k}^{k} f(x - i, y - j) * h(x, y) \tag{4}$$



Obr. 6: Princíp 2D diskrétnej konvolúcie.

4.3.2 Priemerový filter

Pri tejto filtrácií je hodnota pixelu určená priemernou hodnotou pixelov v určitom okolí. Jeho jediným parametrom je teda veľkosť tohoto okolia. Výsledok filtrácie je možné vidieť na obrázku 7.

Voľba veľkosti ovplyvňuje veľkosť detailov, ktoré zostanú zachované. Nemala by teda presahovať veľkost najmenšieho významného detailu v obraze, aby nedochádzalo k jeho chybému označeniu za šum a následnému odstráneniu.

Konvolučná maska pre okolie veľkost masky 3 vyzerá nasledovne :

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \frac{1}{9}$$

(a) Pôvodný obrázok. (b) Filtrovaný obrázok s maskou 3x3. (c) Filtrovaný obrázok s maskou 7x7.

Obr. 7: Aplikácia priemerového filtra.

4.3.3 Mediánový filter

Princípom tohoto nelineárneho filtra spočíva vo výbere mediánu z hodnôt pod maskou. Veľkosť masky je pri tejto filtrácií zásadná. Pri voľbe veľkej masky sú malé detaily úplne potlačené. Táto vlastnosť môže byť využitá v prípade postprocesingu segmentovaného obrazu ak je potrebné odstrániť malé zhluky pixelov. V prípade šumu typu soľ a korenie vykazuje veľmi dobré výsledky. Príklad mediánovej filtrácie je uvedený na obrázku 8.

(a) Pôvodný obrázok. (b) Filtrovaný obrázok s maskou 3x3, $\sigma = 0$. (c) Filtrovaný obrázok s maskou 7x7, $\sigma = 0$.

Obr. 8: Aplikácia mediánového filtra.

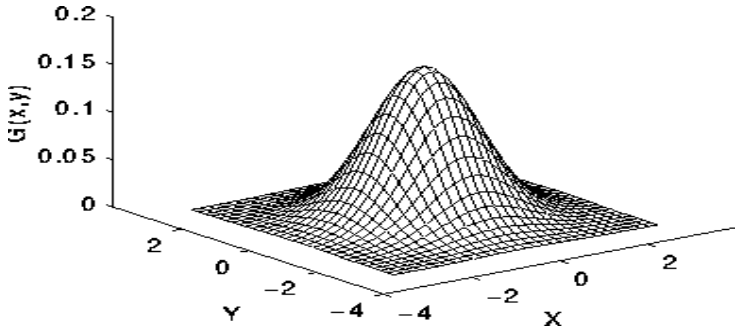
4.3.4 Gaussov filter

Hodnota pixelu pri Gaussovom filtre je určená konvolúciou s maskou, ktorej koeficienty sú určené pomocou vzorca

5.

$$G(x,y) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2+y^2}{\sigma^2}} \tag{5}$$

kde σ^2 je rozptyl, ktorý určuje strmosť Gaussovej funkcie [7], tá je zobrazená na obrázku 9. Na obrázku 10 je možné vidieť výsledok filtrovania Gaussovým filtrom.



Obr. 9: Gausová distribúcia s priemerom v bode [0,0] $\sigma = 1$.

(a) Pôvodný obrázok.

(b) Filtrovaný obrázok s maskou 3x3, $\sigma = 0$.

(c) Filtrovaný obrázok s maskou 7x7, $\sigma = 0$.

Obr. 10: Aplikácia Gaussovho filtra.

4.3.5 Filter typu rozptyl

Tento filter realizuje výslednu hodnotu ako odhad jasu pixelov v okolí [8]. Matematicky je tento filter definovaný pomocou rovnice 6.

$$D\left(i,j\right)=\frac{1}{\left(2n+1\right)^{2}-1}\sum_{p=-n}^{n}\sum_{q=-n}^{n}\left(\bar{f}\left(i,j\right)_{n}-f\left(i+p,j+q\right)\right)^{2} \tag{6}$$

$$\bar{f}\left(i,j\right)_{n}=\frac{1}{\left(2n+1\right)^{2}}\sum_{p=-n}^{n}\sum_{q=-n}^{n}f\left(i+p,j+q\right) \tag{7}$$

kde n predstavuje polomer okolia, $\bar{f}\left(i,j\right)_{n}$ označuje priemerný jas.

Tento filter je zaujímavý v prípade, kedy je potrebné zdôrazniť nehomogenitu obrazu [8].

4.3.6 Bilaterálny filter

Táto technika vznikla v roku 1995 pre vyhladzovanie obrazových dát pri zachovani hrán.

Podstatou metódy je, že okrem filtrovania na základe hodnôt okolitých pixelov berieme do úvahy aj ich vzdialenosti. Tá môže byť chápaná ako geometrická vzdialenosť, a teda pixely sú si blízke ak ležia blízko seba. Ďalšou možnosťou je fonometrická vzdialenosť. V prípade, že majú dva pixely podobnú hodnotu sú si fonometricky blízke.

Táto metóda je tvorená dvoma typmi filtrov a to doménovým a frekvenčným. Doménový filter závisí na geometrickej vzdialenosti, naopak frekvenčný závisý na fonometrickej vzdialenosti. Na obraz sú aplikované oba filtre súčasne.

Jednou z foriem filtrovanie je Gaussovo filtrovanie. V takom prípade filter definvaný vzorcom 8.

$$I(x)=\sum_{n\in N}c\left(n,x\right)s\left(f\left(n\right),f\left(x\right)\right) \tag{8}$$

$$c\left(n,x\right)=e^{-\frac{1}{2}\left(\frac{d(n,x)}{\sigma_{d}}\right)^{2}} \tag{9}$$

$$d\left(n,x\right)=||n-x|| \tag{10}$$

$$s\left(n,x\right)=e^{-\frac{1}{2}\left(\frac{\delta(f(n),f(x))}{\sigma_{r}}\right)^{2}} \tag{11}$$

$$\delta\left(f\left(n\right),f\left(x\right)\right)=||f\left(n\right)-f\left(x\right)|| \tag{12}$$

kde n je pixel okolia a x pixel v strede masky, σ_r fonometrický rozptyl a σ_d geometrický rozptyl.

Princíp spočíva v tom, že v homogénnych oblastiach využíva filter obrazovú informáciu z okolia a využíva doménovu zložku. Ak je v okolí hrana využije sa aj frekvenčná zložka. Homogénne oblasti sú teda zjemnené pričom heterogénne oblasti obsahujúce hrany ostávajú zachované.

Kritickým je nastavenie rozptylov. Väčšia hodnota geometrického rozptylu znamená zosilnenie doménovej zložky čím sa zosilňuje rozmazanie. Naopak väčšia hodnota fonometrického rozptylu sa vplyv frekvenčného filtra znižuje.

Na obrázku 11 je možné vidieť, ako voľba fonometrického rozptylu ovplyvňuje výsledok filtrácie. S vysokou hodnotou fonometrického rozptylu sa filter zachoval ako klasický Gaussov filter. Naopak s nižším fonometrickým rozptylom zachoval hrany a rozmazal len homogénne oblasti.

(a) Pôvodný obrázok. (b) Filtrovaný obrázok : $\sigma_d = 256$, $\sigma_r = 42$. (c) Filtrovaný obrázok : $\sigma_d = 256$, $\sigma_r = 42$.

Obr. 11: Aplikácia bilaterálneho filtra.

4.3.7 Gaborov filter

Gaborov filter predstavi maďarsko-britský fyzik a držiteľ Nobelovej ceny Dennis Gabor. Používa sa pri detekcii frekvencií v rôznych smeroch. Je definovaný ako súčin harmonickej a Gaussovej funkcie. Harmonická funkcia je charakteristická frekvenciou a smerom, Gaussova funkcia ohraničuje harmonickú funkciu [9]. Ďalším mozným použitím je extrakcia výrazných charakteristík, nakoľko je smerovo a frekvenčne nastaviteľný. Matematické vyjadrenie Gáborovho filtra [10]:

$$G(x, y, \theta, f) = exp\left\{-\frac{1}{2}\left[\frac{x_\theta^2}{\delta_x^2} + \frac{y_\theta^2}{\delta_y^2}\right]\right\} cos\left(2\pi f x_\theta\right) \tag{13}$$

$$x_\theta = x\cos\theta + y\sin\theta \tag{14}$$

$$y_\theta = -x\sin\theta + y\cos\theta \tag{15}$$

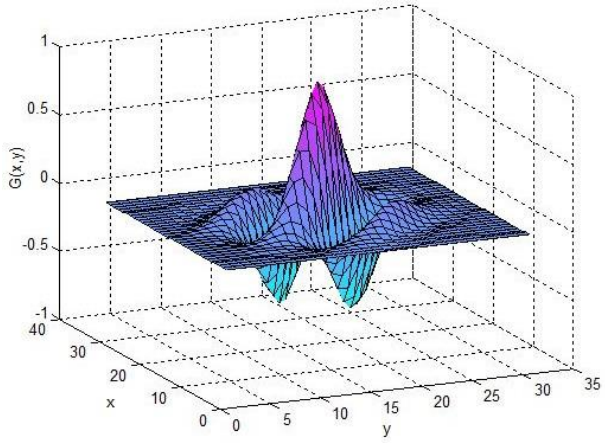
kde θ je orientácia, f frekvencia a δ smerodatná odchylka Gaussovej obálky v smeroch x a y.

Vyfiltrovaný obraz E, ktorý vznikol aplikáciou filtra G na obraz I získame pomocou vzoroca 16 [10].

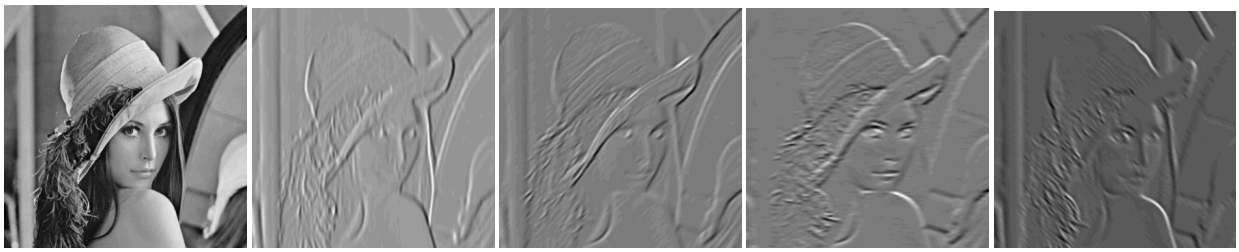
$$E(x,y) = \sum_{u=-\frac{w_x}{2}}^{\frac{w_x}{2}} \sum_{v=-\frac{w_y}{2}}^{\frac{w_y}{2}} G(u,v,O(x,y),F(x,y))\, I(x-u,y-v) \tag{16}$$

kde w_x a w_y predstavujú veľkosť filtra, $O(x,y)$ predstavuje lokálnu orientáciu a $F(x,y)$ lokálnu frekvenciu v bode $[x,y]$.

Priklad Gáborovho filtra je na obrázku 12. Výsledky aplikácie na obrázku 13.



Obr. 12: Gaborov filter. $\sigma = 4$, $\theta = 45°$, $\lambda = 8$, $\delta = 1$ a $\omega = 0$.



(a) Pôvodny obraz. (b) 0° (c) 45° (d) 90° (e) 135°

Obr. 13: Výsledky Gáborovho filtra pre rôzne orientácie.

5 Segmnetácia

Segmentácia obrazu je proces, pri ktorom sa snažíme oddeliť objekt v obraze od pozadia. Pred segmentáciou samotnou je nutné z obrazu odstrániť šum. Metódy pre jeho odstránenie boli predstavené v predchádzajúcej kapitole. V tejto kapitole sú predstavené segmnetačné metódy použité v tejto práci.

5.1 Prahovanie

Prahovanie je jedna z najjednoduchších a najstarších metód segmentacie. Celá podstata tejto segmentačnej techniky tkvie v tom, že sú na základe určitej prahovej hodnoty od seba oddelené pixely pozadia od pixelov objektu. Medzi jej základné výhody patrí jednoduchosť implementácie ako aj jej rýchlosť.

5.1.1 Binárne prahovanie

Úlohou binárneho prahovania je oddelenie pixelov s informačnou hodnotou (pixely objektu) od pixelov pozadia na základe určitej hodnoty. Táto hodnota je označovaná ako T - práh. Výsledkom tejto metódy je obraz, ktorý obsahuje iba dve hodnoty. V prípade šedotónového obrazu sú to hodnoty 0 a 255, v prípade binárneho obrazu hodnoty 0 a 1.

Matematicky je možné zapísať prahovanie ako 17

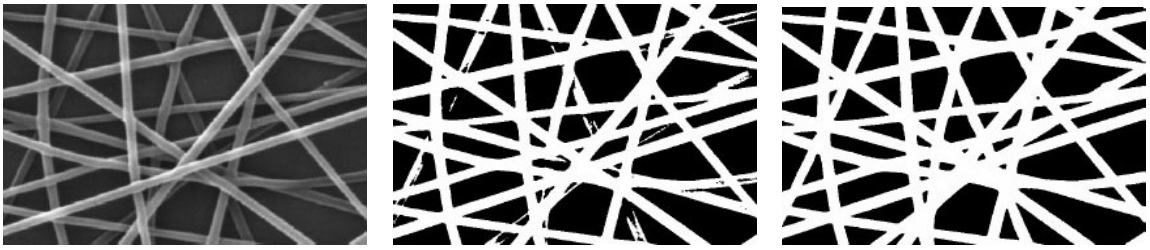
$$f(x) = \begin{cases} 1, & x \geq T \\ 0, & x < T \end{cases} \tag{17}$$

Táto metóda je veľmi jednoduchá, no napriek tomu v určitých prípadoch veľmi spoľahlivá. Problémom je to, že je založená na predpoklade, že pixely objektov a pozadia majú rozdielne úrovne jasu a teda žiaden z pixelov pozadia nemá vyšší jas ako pixel objektu a naopak.

Ďalším problémom je určenie prahu samotného. Ten môže byť nastavovaný ručne, prípadne automaticky. Jednou z metód je percentuálne nastavenie prahu. Ak je známe akú oblasť obrazu zaberajú hľadané objekty, nastaví sa prah tak, aby sa čo najlepšie zhodovala s veľkosťou plochy vysegmentovaných objektov. Ďalšou možnosťou je využitie histogramu. Na obrázku 14 je zobrazený bimodálny histogram. V tomto prípade by sa zvoli prah z oblasti medzi maximami histogramu. Ak sa však v histograme nachádza viacero maxím, môže byť prah chybne zvolený ako hodnota medzi nimi. Na odstránenie tohoto problému je možné použiť pravidlo vzdialenosti d. Za významné maximá teda považujú iba tie, medzi ktorými je vzdialenosť aspoň d jasových úrovní.

Ďalším možným vylepšením je použitie lokálneho prahovania. Ak je obraz nerovnomerne osvetlený, má v rôznych častiach rôzne úrovne jasu a teda rôzne optimálne prahy. Pre vyriešenie tohoto problému sa nepoužije jeden prah pre celý obraz, ale vypočíta sa pre jednotlivé oblasti zvlášť. Na obrázku 15, sú zorbazené výsledky pre lokálne a globálne prahovanie.

Obr. 14: Bimodálny histogram.



(a) Originálny obraz. (b) Globálne prahovanie. (c) Lokálne prahovanie.

Obr. 15: Porovnanie globálneho a lokálneho prahovania.

5.1.2 Otshuho prahovanie

Táto metóda patrí medzi metódy automatického nájdenia prahu. Jej princípom je rozdelenie pixelov do dvoch tried a minimalizácia ich spoločného rozptylu. Prah samotný získame maximalizáciou výrazu 18.

$$\sigma_b^2(T) = n_b(T)\, n_o(T) \left[\mu_b(T) - \mu_o(T)\right]^2 \tag{18}$$

$$n_b(T) = \sum_{i=0}^{T-1} p(i) \tag{19}$$

$$n_o(T) = \sum_{i=T}^{N-1} p(i) \tag{20}$$

$$\mu_b(T) = \sum_{i=0}^{T-1} \frac{ip(i)}{n_b(T)} \tag{21}$$

$$\mu_o(T) = \sum_{i=T}^{N-1} \frac{ip(i)}{n_o(T)} \tag{22}$$

Kde $\sigma_b^2(T)$ je vzájomný rozptyl oboch skupín bodov, T je aktuálne skúmaná hodnota prahu a $[0, N\check{}1]$ je rozsah úrovní jasu. $p(i)$ je počet pixelov s danou intenzitou. n_b predstavuje počet pixelov pozadia, n_o počet pixelov objektu. μ_b a μ_o predstavujú priemer pixelov pozadia a objektu.

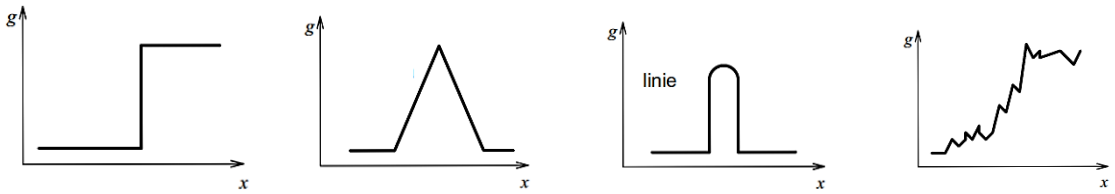
Ak využívame histogram, nemusíme skupiny prepočítavať znova a znova. Stačí iba prvotné rozdelenie a pre každý nový prah len presúvame príslušný počet pixelov získaných z histogramu.

5.2 Detekcia hrán pomocou prvej derivácie

Predchádzajúce metódy pracovali na princípe oddelenia pixelov pozadia od pixelov objektu na základe prahovej hodnoty. Metódy detekcie hrán nedetegujú objekt samotný, ale iba miesta, kde dochádza k významným zmenám jasu – hrany. Výsledkom teda nie je segmentovaný objekt samotný, ale mapa hrán. Typické hranové profily sú uvedené na obrázku 16.

Prvé tri sú ideálne hrany, s ktorými sa v reálnych podmienkach moc nestretávame. Bežný profil je na obrázku 16d. Práve šum spôsobuje problémy s detekciou hrán. O eliminácii šumu pojednáva kapitola 4.3.

Ďalším problémom je fakt, že ku zmene intenzity môže dochádzať v širokej oblasti. Za hranu môžeme považovať celú oblasť, prípadne len jej stred.



(a) Skoková. (b) Strechová. (c) Línia. (d) Zašumená.

Obr. 16: Typické hranové profily.

Pri hľadaní hrán pomocou prvej derivácie využívame fakt, že v homogénnej oblasti je prvá derivácia nulová, a teda v miestach kde je vysoká sa nachádza hrana [7]. Tieto metódy sa taktiež nazývajú gradientné metódy.

5.2.1 Gradient

Gradient je výsledkom derivácie obrazu v smere x a y. Je to vektor, ktorý je definovaný ako vektor funkcie dvoch premenných $f(x, y)$. Získame ho pomocou vzorca 23. Jeho veľkosť je definovaná pomocou rovnice 24 a smer pomocou rovnice 25.

$$\nabla f = \begin{bmatrix} G_x \\ G_y \end{bmatrix} = \begin{matrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{matrix} \tag{23}$$

$$mag\left(\nabla f\right) = \sqrt{G_x^2 + G_y^2} = sqrt\frac{\partial f}{\partial x} + \frac{\partial f}{\partial y} \tag{24}$$

$$\Phi\left(x, y\right) = \arctan\frac{G_y}{G_x} \tag{25}$$

Smer gradientu môžeme využiť pri určení smeru hrany, keďže gradient je na hranu kolmý. Odhad derivácií G_x a G_y v diskrétnom obraze je možné určiť rozdielom dvoch susedných hodnôt. V [7] je uvedený symetrický variant daný vzorcom 26 pre G_x a vzorcom 27 pre G_y

$$G_x \approx f(x+1, y) - f(x-1, y) \tag{26}$$

$$G_y \approx f(x, y-1) - f(x, y-1) \tag{27}$$

5.2.2 Hranové operátory

V predchádzajúcej časti bol vysvetlený gradient. Táto časť pojednáva o operátoroch, ktoré sa používajú pre výpočet jednotlivých zložiek gradientu pomocou ich konvolúcie s obrazom.

Existuje niekoľko operátorov (konvolučných jadier) pre hľadanie hrán založených na princípe gradientu. Napríklad :

- Robertsov operátor

$$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

- Prewittovej operátor

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

- Sobelov operátor

$$\begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

- Scharrov operátor

$$\begin{bmatrix} 3 & 0 & 3 \\ 10 & 0 & -10 \\ 3 & 0 & -3 \end{bmatrix} \begin{bmatrix} 3 & 10 & 3 \\ 0 & 0 & 0 \\ -3 & -10 & -3 \end{bmatrix}$$

Všetky operátory boli uvedené pre výpočet G_x a G_y.

Postup je teda nasledovný. Vypočítame x-ovú a y-ovú zložku gradientu g_x a g_y. Následne pre každý bod vypočítame veľkosť gradientu $|g| = \sqrt{G_x^2 + G_y^2}$. Po získaní veľkostí gradientov môžeme využiť prahovanie, aby sme potlačili gradienty s malou odozvou.

5.2.3 Kompasové hranové operátory

Tieto operátory fungujú na podobnom princípe ako predchádzajúce. Rozdiel je v tom, že pri kompasových operátoroch vykonávame rotácie po 45°. Výsledný gradient predstavuje maximálny gradient získany jednotlivými maskami.

- Kirschov operátor

$$\begin{bmatrix} 5 & 5 & 5 \\ -3 & 0 & -3 \\ -3 & -3 & -3 \end{bmatrix} \begin{bmatrix} -3 & 5 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & -3 \end{bmatrix} \begin{bmatrix} -3 & -3 & 5 \\ -3 & 0 & 5 \\ -3 & -3 & 5 \end{bmatrix} \begin{bmatrix} -3 & -3 & -3 \\ -3 & 0 & 5 \\ -3 & 5 & 5 \end{bmatrix}$$

- Nevatia Babu

$$\frac{1}{1000}\begin{bmatrix} 100 & 100 & 0 & -100 & -100 \\ 100 & 100 & 0 & -100 & -100 \\ 100 & 100 & 0 & -100 & -100 \\ 100 & 100 & 0 & -100 & -100 \\ 100 & 100 & 0 & -100 & -100 \end{bmatrix} \frac{1}{1102}\begin{bmatrix} 100 & -32 & -100 & -100 & -100 \\ 100 & 78 & -92 & -100 & -100 \\ 100 & 100 & 0 & -100 & -100 \\ 100 & 100 & 92 & -78 & -100 \\ 100 & 100 & 100 & 32 & -100 \end{bmatrix}$$

$$\frac{1}{1102}\begin{bmatrix} -100 & -100 & -100 & -100 & -100 \\ 32 & -78 & -100 & -100 & -100 \\ 100 & 92 & 0 & -92 & -100 \\ 100 & 100 & 100 & 78 & -32 \\ 100 & 100 & 100 & 100 & 100 \end{bmatrix} \frac{1}{1000}\begin{bmatrix} -100 & -100 & -100 & -100 & -100 \\ -100 & -100 & -100 & -100 & -100 \\ 0 & 0 & 0 & 0 & 0 \\ 100 & 100 & 100 & 100 & 100 \\ 100 & 100 & 100 & 100 & 100 \end{bmatrix}$$

$$\frac{1}{1102}\begin{bmatrix} -100 & -100 & -100 & -100 & -100 \\ -100 & -100 & -100 & -78 & 32 \\ -100 & -92 & 0 & 92 & 100 \\ -32 & 78 & 100 & 100 & 100 \\ 100 & 100 & 100 & 100 & 100 \end{bmatrix} \frac{1}{1102}\begin{bmatrix} -100 & -100 & -100 & -32 & 100 \\ -100 & -100 & -92 & 78 & 100 \\ -100 & -100 & 0 & 100 & 100 \\ -100 & -78 & 92 & 100 & 100 \\ -100 & 32 & 100 & 100 & 100 \end{bmatrix}$$

Ako kompasové filtre môžeme použiť aj operátory z predchádzajúcej časti. Kirschov operátor je uvedený iba pre smery S, SV, V a JV. Nevatia Babu operátor predstavuje výnimku a to tým, že je rotovaný po 30°. Aj v tomto prípade je uvedená len polovica smerov a to v rozsahu 0° – 150°.

5.2.4 Frei Chen

Frei chen detektor pracuje s deviatimi jadrami o veľkosti 3x3. Na rozdiel od predchádzajúcich operátorov používa 9 unikátnych masiek. Tieto masky sú :

$$\frac{1}{2\sqrt{2}}\begin{bmatrix}1 & \sqrt{2} & 1\\ 0 & 0 & 0\\ -1 & -\sqrt{2} & -1\end{bmatrix}\ \frac{1}{2\sqrt{2}}\begin{bmatrix}1 & 0 & -1\\ \sqrt{2} & 0 & -\sqrt{2}\\ 1 & 0 & -1\end{bmatrix}\ \frac{1}{2\sqrt{2}}\begin{bmatrix}0 & -1 & \sqrt{2}\\ 1 & 0 & -1\\ -\sqrt{2} & 1 & 0\end{bmatrix}\ \frac{1}{2\sqrt{2}}\begin{bmatrix}\sqrt{2} & -1 & 0\\ -1 & 0 & 1\\ 0 & 1 & -\sqrt{2}\end{bmatrix}$$

$$\frac{1}{2}\begin{bmatrix}0 & 1 & 0\\ -1 & 0 & -1\\ 0 & 1 & 0\end{bmatrix}\ \frac{1}{2}\begin{bmatrix}-1 & 0 & 1\\ 0 & 0 & 0\\ 1 & 0 & -1\end{bmatrix}\ \frac{1}{6}\begin{bmatrix}1 & -2 & 1\\ -2 & 4 & -2\\ 1 & -2 & 1\end{bmatrix}\ \frac{1}{6}\begin{bmatrix}-2 & 1 & -2\\ 1 & 4 & 1\\ -2 & 1 & -2\end{bmatrix}$$

$$\frac{1}{3}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

Prvé štyri masky sú použité pre detegovanie hrán, ďalšie štyri pre detegovanie línií a posledná pre vypočítanie priemeru.

Pre detekciu hrán sa používa rovnica 28.

$$\cos e = \sqrt{\frac{M}{S}} \tag{28}$$

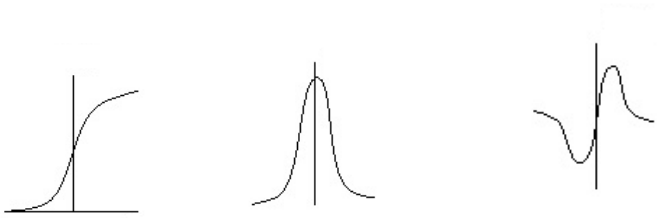
$$M = \sum_{k=1}^{4} (G_k * I)^2 \tag{29}$$

$$S = \sum_{k=1}^{9} (G_k * I)^2 \tag{30}$$

$$\tag{31}$$

5.3 Detekcia hrán pomocou druhej derivácie

Druhá derivácia predstavuje rýchlosť zmeny jasu – zmenu zmeny. Oproti metódam prvej derivácie nezískame informáciu o smere hrany, preto tieto metódy nie sú vhodné ak túto informáciu potrebujeme. Na obrázku 17 môžeme vidieť, že v mieste maxima prvej derivácie prechádza druhá derivácia nulou.



(a) Pôvodny funkcia. (b) 1. derivácia. (c) 2. derivácia.

Obr. 17: Príklad funkcie a jej derivácií.

Pri hľadaní hrán musíme teda vypočítať druhú deriváciu a následne určiť miesta prechodu nulou. V diskrétnom obraze je možné druhú deriváciu vypočítať pomocou vzorca 32 a 33.[7]

$$\frac{\partial^2 f(x,y)}{\partial x^2} \approx [f(x+1,y) + (x-1,y)] - 2f(x,y) \tag{32}$$

$$\frac{\partial^2 f(x,y)}{\partial y^2} \approx [f(x,y+1) + (x,y-1)] - 2f(x,y) \tag{33}$$

Rovnako ako v prípade prvej derivácie je aj druhú deriváciu možné vypočítať pomocou konvolúcie s hranovým operátorom pre druhú deriváciu.

5.3.1 Laplaceov operátor

Laplaceov operátor patrí medzi operátory počítajúce druhú deriváciu. Vyznačuje sa tým, že súčet všetkých jeho prvkov je rovný nule a kladie dôraz na stredové body. Tento operátor je však citlivý na šum a môže spôsobiť nájdenie dvojitých hrán. Príklad Laplaceovho operátora :

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 4 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

5.3.2 Laplacian of Gaussian (LoG)

Pomocou konvolúcie obrazu a Gaussovho filtra je možné docieliť rozmazanie obrazu a tým by sa znížila odozva šumu na konvolúciu s Laplaceovým operátorom. Tento výsledok je možné docieliť jednou operáciou (nemusíme teda najskôr aplikovať Gaussov filter a následne Laplaceov operátor). Druhá derivácia Gaussovho filtra je daná vzorcom 34.

Veľkou výhodou LoG operátora je možnosť voľby veľkosti jadra. Čím väčšie jadro použijeme, tým väčšiu odolnosť voči šumu dosiahneme. Na druhú stranu by malo byť jadro tak veľké ako najmenší detail, ktorý chceme zachytiť. Veľkosť jadra taktiež zvyšuje časovú náročnosť výpočtu.

$$G''(x,y) = \frac{x^2+y^2-\sigma^2}{\sigma^4} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{34}$$

5.3.3 Difference of Gaussian (DoG)

Pri tomto postupe sa obraz rozmaže dvakrát Gaussovým filtrom. Raz s väčším σ a raz s menším. Tieto dva obrazy od seba následne odčítame. Výsledný obraz je teda daný vzorcom 35.

$$DoG = G_{\sigma_1} - G_{\sigma_2}, kde \sigma_1 < \sigma_2 \tag{35}$$

Podľa práce Marra a Hildertha [11] dosiahneme aproximáciu LoG pri pomere $\frac{\sigma_2}{\sigma_1} = 1.6$.

5.3.4 Nájdenie prechodu nulou

Ako bolo spomínané v kapitole o detekcii hrán pomocou druhej derivácie je nutné nájsť miesta, kde druhá derivácia prechádza nulou. Takto vytvoríme hľadanú mapu hrán. Jedna z možností je prejdenie obrazu bod po bode a všetky pixely, kde druhá derivácia dosahuje hodnotu 0 označiť za pixely hrany. Tento postup však nie je vhodný, pretože nulovú hodnotu druhej derivácie majú aj pixely v homogénnych oblastiach . S ďalším problémom sa môžeme stretnúť ak druhá derivácia prechádza nulou práve medzi dvoma pixelmi. Tieto pixely nemajú síce nulovú hodnotu druhej derivácie, napriek tomu sú pixelmi hrany. Tento problém môžeme odstrániť použitím masky 2x2, ktorú použijeme na vyhľadanie bodov, v ktorých došlo k zmene znamienka.

$$\begin{bmatrix} \mathbf{a} & b \\ c & d \end{bmatrix}$$

Za stred masky určíme prvok **a**. Ak sa líši jeho znamienko od iného prvku v maske je tento bod označený za hranový.

Túto masku je možné použiť k určitej forme prahovania. Ak majú **a** a napríklad **b** rôzne znamienka, tak je bod **a** označený za bod hrany iba v prípade ak $|a-b| > T$, kde T je hodnotou prahu.

5.4 Cannyho detektor hrán

Na začiatku 80. rokov J. F. Canny stanovil nasledujúce vlastnosti ideálneho hranového detektora:

- **Minimálna chyba detekcie hrán** – všetky dôležité hrany musia byť detegované a nesmie byť detegovaná žiadna falošná hrana.
- **Správna lokalizácia** – vzdialenosť medzi skutočnou a detegovanou hranou musí byť čo najmenšia.
- **Iba jedna odozva** – každá hrana musí byť detegovaná iba raz.

Tento detektor je môžeme zaradiť medzi metódy využívajúce prvú deriváciu, pretože pre detekciu hrán je nutné poznať smer aj veľkosť gradientu.

Najskôr sa na obraz aplikuje filter pre redukciu šumu. Následne sa vypočíta prvá derivácia podľa osi x a y. Z týchto hodnôt získame veľkosť aj smer gradientu. Ďalším krokom je stenšenie hrán. Na to využijeme metódu *Nonmaxima suppression*. Na výsledny obraz sa aplikuje hysterzne prahovanie.

Týmto krokom sa snažíme eliminovať nevýznamné a falošné hrany. Vstupom tejto metódy sú prahy T_1 a T_2, kde $T_1 < T_2$. V obraze ostanú iba tie hrany, ktoré majú gradient väčší ako T_2 (silné hrany) a tie hrany, ktoré majú gradient väčší ako T_1 (slabé hrany) a zároveň sú pripojené aspoň k jednej silnej hrane.

5.4.1 Nonmaxima suppression

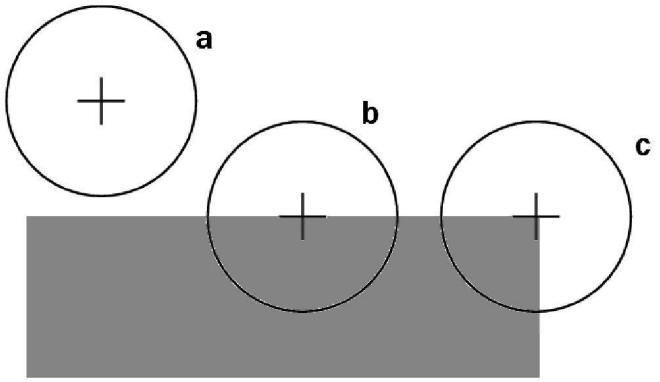
V predchádzajúcej časti bola spomenutá táto metóda. Jedná sa o poloprahovaciu metódu, ktorá je založená na predpoklade, že hrana dáva najväčšiu odozvu v mieste, kde sa skutočne nachádza.

Pre každý bod hrany vyberieme susedné body hrany v smere gradientu. Následne ich hodnoty porovnáme. Ak je hodnota skúmaného bodu menšia ako hodnota ľubovolného bodu z porovnávaných, je skúmaný bod označený za neplatný. Týmto postupom sa snažíme splniť podmienku správnej lokalizácie hrany ideálneho detektora.

Veľkou nevýhodou je, že v mieste kde sa stretávajú tri hrany dôjde k ich rozpojeniu, pretože gradient jednej z nich bude vždy väčší ako gradienty ostatných.

5.5 SUSAN

V roku 1997 publikovali S.M. Smith a J.M. Brady metódu SUSAN (Smallest Univalue Segment Assimilating Nucleus) [12]. Tá využíva kruhovú masku pomocou ktorej vypočítame odozvu na detektor. Príklad masky je uvedený na obrázku



Obr. 18: Príklad použitia masky. (a) bez odozvy. (b) veľká odozva – hrana. (c) veĺká odozva – roh.

Pre určenie veľkosti odozvy detektoru použijeme vzťah 38.

$$R(r_0) = \begin{cases} \frac{3}{4}n_{max} - n(r_0), & \frac{3}{4}n_{max} > n(r_0) \\ 0, & inak \end{cases} \tag{36}$$

$$n(r_0) = \sum_r c(r, r_0) \tag{37}$$

$$c(r, r_0) = \exp^{-\left(\frac{I(r)-I(r_0)}{t}\right)^6} \tag{38}$$

kde r_0 je stred masky, $n(r_0)$ je cena bodu, r je bodom masky, $I(r)$ je intenzita jasu v bode r, t určuje množstvo potlačeného šumu a veľkosť detegovaných hrán a n_{max} je konštanta predstavujúca maximálnu cenu, ktorú je možné získať.

V prípade, že chceme určiť smer hrany použijeme rovnicu 39 pre vypočítanie bodu CoG v maske. Hľadaný smer je kolmý na vektor medzi CoG a r_0.

$$CoG(r_0) = \frac{\sum_r rc(r, r_0)}{\sum_r c(r, r_0)} \tag{39}$$

kde r sú body priloženej masky.

SUSAN patrí medzi nelineárne filtre a statistické segmentačné metódy. Je odolný voči šumu a schopný detegovať rohy. Bohužiaľ je aj výpočetne náročný.

5.6 K-Means segmentácia

K-means patrí k najzákladnejším metódam pre zhlukovanie. Zhlukovanie je postup, pri ktorom sa snažíme hľadať súvislosti v celku a triediť objekty do zhlukov na základe podobných vlastností. Túto vlastnosť môžeme využiť pri segmentácií, keďže predpokladáme, že pixely objektu sú si navzájom podobnejšie medzi sebou ako s pixelmi pozadia.

V prvom rade je nutná inicializácia centroidov. Centroid je bod, ktorý reprezentuje zhluk. Na základe vzdialenosti od tohoto bodu priradzujeme bod do zhluku. V prípade štandardnej implementácie sa tieto body volia náhodne, čo vedie k rôznym výsledkom. Aby sme eliminovali tento efekt sú hodnoty inicializované na uživateľom zadanú hodnotu.

Ďalším krokom je určenie hodnoty pixelu. Tú určíme ako priemernú hodnotu pixelov v okolí. Matematicky teda pomocou vzorca 40.

$$V(x,y) = \frac{1}{(2n+1)} \sum_{p=-n}^{n} \sum_{q=-n}^{n} I(x+p, y+q) \tag{40}$$

kde $I(x,y)$ predstavuje intenzitu jasu v bode x, y a n predstavuje polomer okolia.

Následne je vypočítaná vzidalenost medzi bodmi obrazu a všetkými clustermi. Bod sa pridá do clustra, ktorého centroid je k nemu najbližšie. Pre výpočet vzdialenosti môže byť použitá napríklad euklidovská vzdialenosť definovaná ako :

$$d(x,y) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2} \tag{41}$$

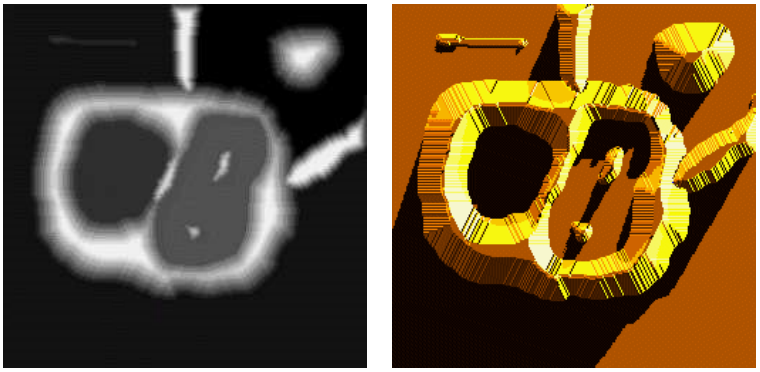
Po priradení všetkých bodov sú centroidy znova prepočítané. Následne znova zistíme vzdialenosti všetkých bodov od centroidov. Ak žiaden bod nezmení svoj zhluk algoritmus končí.

Prepočítavanie pozície centroidov môžeme vykonávať aj pri každej zmene zhluk (pridanie alebo odobranie pixelu).

Výsledkom je binárny obraz, kde sú pixely rozdelené na základe príslušnosti do zhlukov.

5.7 Watershed

Táto metóda vychádza z myšlienky, že sa na obraz pozeráme ako na topografický reliéf. Tento reliéf postupne zaplavíme vodou. Krajinu môžeme zaplaviť ponorením do vody, prípadne padajúcimi kvapkami. Krajina je rozdelená do povodí, postupne sa tieto povodia rozrastajú až pokým sa nestretnú.V miestach, kde sa stretnú dve povodia vznikne hrádza [13]. Na obrázku 19 je vidieť ako metóda chápe obraz.



(a) Pôvodný obraz. (b) Toptografický reliéf.

Obr. 19: Transformácia obrazu na topografický reliéf.

Pri prevode na reliéf je úroveň intenzity jasu pixelu chápaná ako jeho výška v teréne. Každý pixel má definovaných susedov. Podľa toho, či berieme do úvahy 4–susedstvo (pixel nad,pod,vľavo a vpravo od pixelu) alebo 8–susedstvo (pridáme diagonálne smery) sa môžu výsledky metódy líšiť. Ďalším dôležitým pojmom je *sigulárne minimum*. Sú to pixely, ktorých susedia nie sú vyšší ako skúmany pixel. *Minimálne plošina* je plošina, z ktorej je možné zostúpiť. Naopak *Non-minimálne plošina* je tvorená miestami, z ktorých sa zostúpiť nedá.

Myšlienku algoritmu zaviedol S. Beucher a C. Lantuéjoul [14]. Najskôr je obraz prevedený na reliéf. Následne určíme minimá. Z týchto miním začneme zaplavovať terén. Ak sa stretnú dve povodia, vznikne medzi nimi hrádza. Algoritmus končí v momente, keď je zaplavený celý terén.

5.7.1 Vincent Soille algoritmus

Hlavným rysom tohoto algoritmu je presegmentovanie malými segmentami. Pozostáva z dvoch krokov. Najskôr sú pixely zpradené v závislosti na zvyšujúcej sa intenzite jasu. Druhým krokom je samotný zaplavovací proces. Ten začína v minimách. Každému minimu je priradený *label*. Ide o akúsi značku, ktorá slúži ako identifikátor povodia. Pri zaplavovaní je každému pixelu v okolí nejakého povodia, ktorý má úroveň h priradený label daného povodia. Ak má označený pixel viac ako jedno povodie, je označený za hrádzu. Pixely, ktoré majú na konci tejto časti stále label povodia, sa použijú ako nové minimá pre lebel h, ich pripojené komponenty dostanú nový label [15].

5.7.2 Dážď

Táto technika zaplavuje terén zhora. Vychádza z predstavy, že na terén prší, kvapky ďažďa padajú vždy najstrmšou cestou. V najstrmších miestach sa teda nachádzajú regionálne minimá.

5.7.3 Mayerov algoritmus

Tento algoritmus začína zaplavovací proces na okraji obrazu. Na začiatku sa vytvoria **markery**, koré predstavujú body, kde začne záplava. Každý z nich má svoj *label*. Susedné pixely označených pixelov sú vložené do prioritnej fronty. Ich priorita je daná ich úrovňou jasu. Pixel s najvyššou prioritou je zo fronty vybraný. Ak sú susedia tohoto pixelu označený rovnakým labelom, je týmto labelom označený aj skúmany pixel. Do fronty sa pridajú všetci jeho neoznačený susedia. Tento proces opakujeme pokým nie je fronta prázdna.

6 Implementácia

V tejto časti budú popísané vzorky praktickej realizácie modulu. Ten vychádza z teórie popísanej v predchadzajúcich kapitolách. Cieľom je rožšírenie systému FOTOM NG o modul pre segmentáciu pór.

6.1 FOTOM NG

Systém FOTOM NG je založený na modulárnej platforme NetBeans Platform. Na tejto platforme je taktiež založené vývojové prostredie NetBeans. Táto platforma je implementovaná v jazyku Java a poskytovaná pre tvorbu komplexných aplikácií, ktoré využívajú modulárne rožšírenia. Tento systém predstavuje komplexný nástroj pre spracovanie a analýzu snímkov. Jeho vývoj začal v roku 2008 na Fakulte elektrotechniky a informatiky VŠB–TU Ostrava. Vhľadom na fakt, že je založený na platforme NetBeans Platform, je kladený veľký dôraz na modularitu, čo vedie k jednoduchému integrovaniu nových funkcií.

6.2 Vývojove a testovacie prostredie

Vývoj a testovanie modulu prebehlo na nasledujúcej zostave softwaru a hardwaru :

- Operačný systém Windows 10 Professional 64 bit
- Procesor Intel Pentium N3540 (2.16 GHz, 4 jadrá);
- Grafická karta NVIDIA GeForce GT 920M (pamäť 1 GB)
- Operačná pamäť 4GB DDR3
- JDK vo verzií 1.8.0
- NetBeans vo verzií 8.2

6.3 Modul segmentácie

Modul bol navrhnutý na základe požiadavku v konkrétnom projekte, ktorý bol realizovaný v spolupráci s Fakultou materiálovo–technologickou VŠB–TU Ostrava. Cieľom projektu je možnosť analýzy vnútornej štruktúry materiálov, vďaka čomu je možné porovnať vhodnosť zvolených materiálov ako aj postup výroby konkrétneho tvaru. Táto práca predstavuje prvú časť projektu a to samotnú segmentáciu kovov. Výstup tohoto programu je použitý na vytvorenie 3D modelu vzorky a výpočet objemov materiálov. Tento modul je však možné použiť na segmentáciu ľubovolných obrazov. Modul je v systéme FOTOM NG implementovaný ako modul **fotom–metal**.

Základná štruktúra modulu je nasledovná :

- Nahratie dát

- Úprava histogramu (volitelná)
- Filtrácia (volitelná)
- Segmentácia (volitelná)
- Aplikovanie metód na všetky snímky
- Uloženie upravených snímkov vo formáte png

6.4 Nahratie obrazu

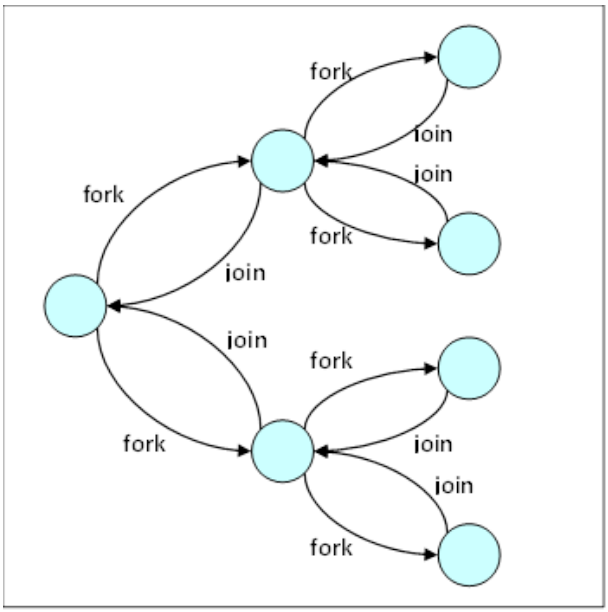
Modul v sučasnosti podporuje základne formáty ako png, jpg a iné. Vzhľadom na to, že všetky testovacie data boli dodané vo formáte DICM bol vytvorený pre tento modul aj základný DICOM konvertor. Ten je reprezentovaný triedou **DICOMReader**. Výsledkom konverzie je iba obraz samotný. Ako bolo spomenuté v kapitole 3, súbory DICOM obsahujú aj mnoho iných informácií. Tie ale nie sú pre potreby tohoto modulu potrebné, preto sa ďalej nepoužívajú.

Samotný zoznam obrazov reprezentuje trieda **ImageList**. V nej je uložený zoznam súborov, ktoré sa nachádzajú vo zvolenej zložke a ktoré majú validný formát. Táto trieda vykonáva základné manipuácie (ďalší obraz, predchádzajúci obraz) ako aj konverziu na triedu **Image**.

Trieda **Image** reprezentuje obraz samotný. Obrazové data sú uložené vo formáte jednorozmerného celočíselného poľa. V celom moduly sa pracuje ďalej najmä s touto reprezentáciou. Vzhľadom na potreby kopírovania dát podporuje táto trieda paralelné kopírovanie pomocou frameworku **Fork Join**.

6.5 Fork Join Framework

Tento framework predstavuje rožšírenie štandartnej kolekcie tried paralelného programovania v jazyku Java pridaním nových techník paralelizácie. Toto rožšírenie je štandartnou súčasťou Java Runtime Envoroment od verzie 7. Jednou z techník, ktoré tento framework pridáva je for-join model. Princípom tohoto modelu je rekurzívne rozdelenie úlohy na menšie časti (fork). Tieto časti sú následne spracované a ich výsledky spätne spojené do jedného celku (join). Schému tohoto modelu je možné vidieť na obrázku 20.



Obr. 20: Schéma modelu Fork-Join.

V momente, kedy je úloha rozdelená na menšie časti, je nutné zvoliť algoritmus, ktorý jednotlivé časti spracuje. V tomto frameworku je tento problém riešený pomocou work stealing algoritmu [16]. Ten je založený na spolupráci vlákien, ktoré spracuvávajú jednotlivé časti. Každé z nich má frontu úloh, ktoré chce obslúžiť. Úlohy predstavujú časti, ktoré vznikli rozdelením pôvodnej úlohy. V momente, keď vlákno vyprázdni svoju frontu, prevezme časť úloh z fronty iného vlákna, ktoré ešte svoju frontu nevyprázdnilo. Snahou tohoto princípu je čo najrýchlejšie paralelné spracovanie úloh.

Tento framework je spolu so základným použitím vlákien použitý v každom filtračnom aj segmentačnom algoritme. Cieľom bolo čo najviach urýchliť spracovanie obrázkov a tým dosiahnuť čo najlepší uživateľský zážitok.

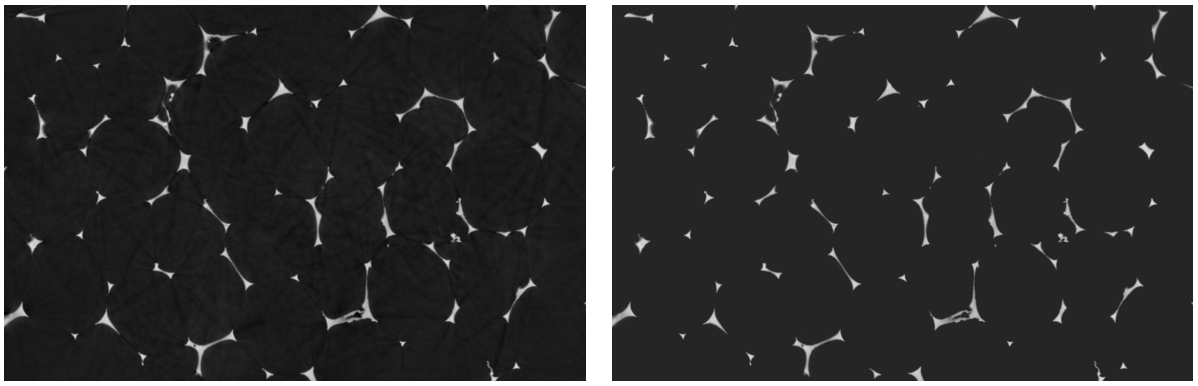
6.6 Operácie úpravy histogramu

Pod pojmom operácie s histogramom sú zahrnuté tri základné operácie a to :

- Histogramová ekvalizácie
- Nastavenie minimálnej úrovne jasu
- Nastavenie maximálnej úrovne jasu

Nastavenie minimálnej úrovne jasu predstavuje operáciu, kedy každý pixel, ktorého úroveň jasu je nižšia ako uživateľom definovaná hodnota, je nahradený pixelom s touto definovanou hodnotou úrovne jasu. Rovankým spôsobom prebieha úprava maximálnej úrovne jasu, s tým rozdielom že hodnota nesmie byť vyššia ako uživateľom definovaná.

Na obrázku 21 je možné vidieť, ako pomocou úpravy minimálnej hodnoty jasu na hodnotu 37 upravíme čierne oblasti obrazu. V týchto oblastiach sa na pôvodnom obraze nachádzajú artefakty, ktoré vznikli pri snímaní obrazu.

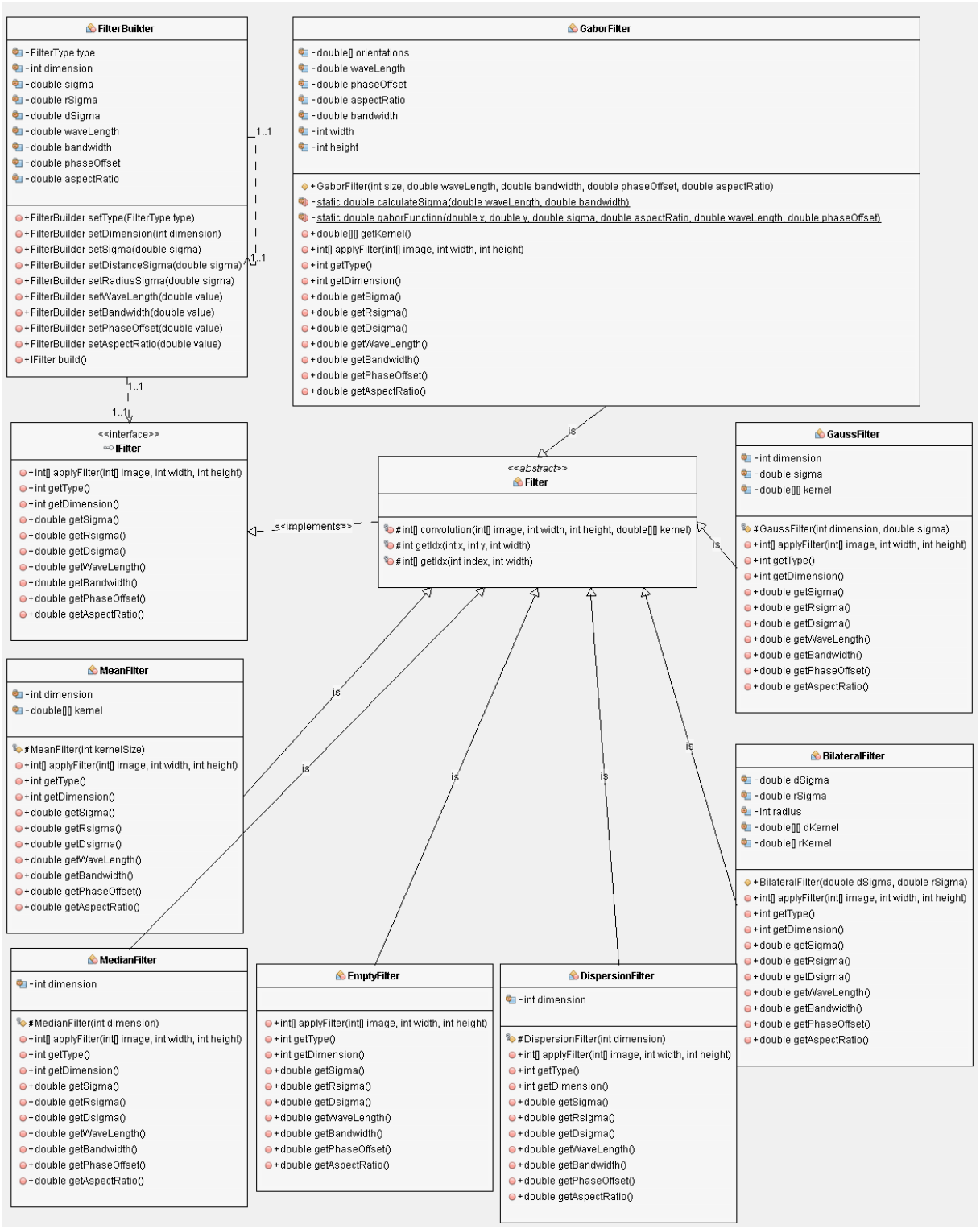


(a) Pôvodný obraz. (b) Upravený obraz.

Obr. 21: Príklad použitia úpravy histogramu.

6.7 Filtrácie

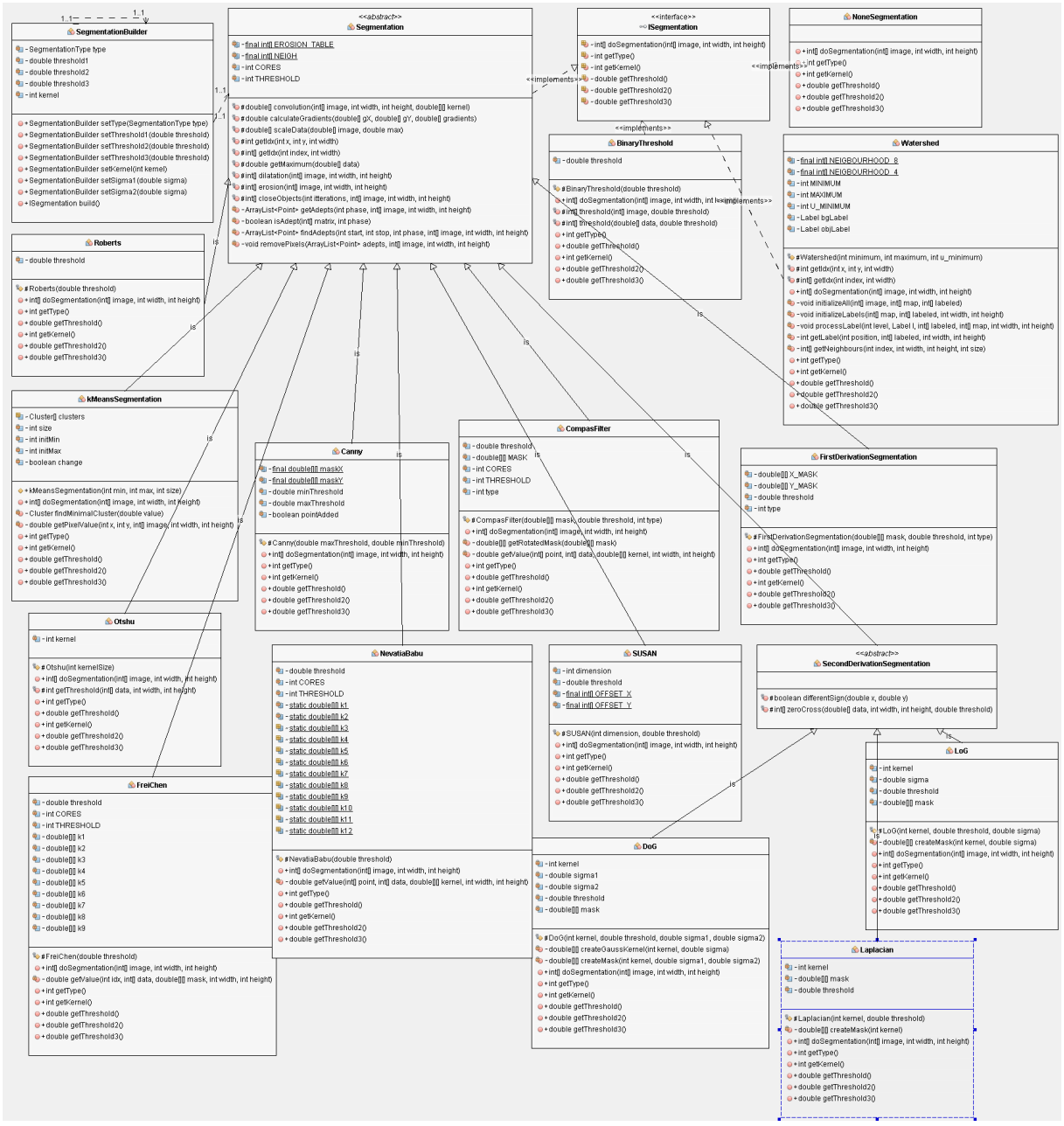
Výsledný modul obsahuje implementácie všetkých filtračných metód popísaných v kapitole 4.3. Uživatel má teda možnosť zvoliť si filter, ktorý chce použiť a samozrejme môže modifikovať parametre týchto filtrov. Všetky triedy reprezentujúce filtračné metódy implementujú rozhranie **IFilter**, prípadne dedia z tried, ktoré toto rozhranie implementujú. V ostatných častiach modulu sa pracuje výhradne s týmto rozhraním. Tento postup unifikuje komunikáciu medzi filtračnými metódami a zvyškom aplikácie. To umožnuje jednoduché pridanie nových filtrov bez nutnosti väčšieho zásahu do vnútornej štruktúry. Na obrázku 22 je uvedený triedny diagram, ktorý popisuje implementáciu filtrov.



Obr. 22: Triedny diagram popisujúci implementáciu filtrov.

6.8 Segmentácie

Rovnako ako v prípade filtračných metód aj v prípade segmentačných metód obsahuje modul všetky, ktoré boli opísané v kapitole 5. Ako v prípade filtrov, aj v tejto časti bol kladený dôraz na modularitu a ľahkú rozšíritelnosť implementovaných metód. Preto aj v tomto prípade všetky triedy reprezentujúce filtračné metódy implementujú rozhranie **ISegmentation** prípadne dedia z tried, ktoré toto rozhranie implementujú. Na obrázku 23 je uvedený triedny diagram, ktorý popisuje implementáciu segmentačných metód.



Obr. 23: Triedny diagram popisujúci implementáciu segmentačných metód.

Problémom, s ktorým som sa stretol pri riešení tejto časti bol fakt, že všetky segmentačné metódy založené na hľadaní mapy hrán nie sú použitelné pre potreby ďalšieho spracovania snímkov. To znamená, že som musel navrhnúť postup, pomocou ktorého získame z mapy hrán pixely objektu.

Pri hľadaní riešenia som narazil na postup, kde je obrazom vedená zľava do prava priamka. Následne sa prechádzame každým pixelom ležiacim na priamke. Ak je počet priesečnikov priamky a hrán vpravo od skúmaneho bodu nepárny, znamená to, že sa jedná o bod objektu.

Pri testovaní však táto metóda neposkytovala uspokojivé výsledky. Často totiž viedla hrana rovnobežne s priamkou, čo zapríčinilo chybné označenie bodu. Následne sa táto chyba prejavila vo zvyšnej časti riadku. Ďalším problémom bol fakt, že ak objekt ležal na okraji obrazu, nebol z tejto strany ohraničený hranou (nedochádzalo k zmene intenzity). To zapríčinilo chýbajúcu hranu do výpočtu a výsledná priamka bola invertovaná (pixely pozadia boli označené za pixely objektu a naopak).

Z tohoto dôvodu som sa rozhodol pre komplexnejšie riešenie.

6.9 Extrakcia komponent

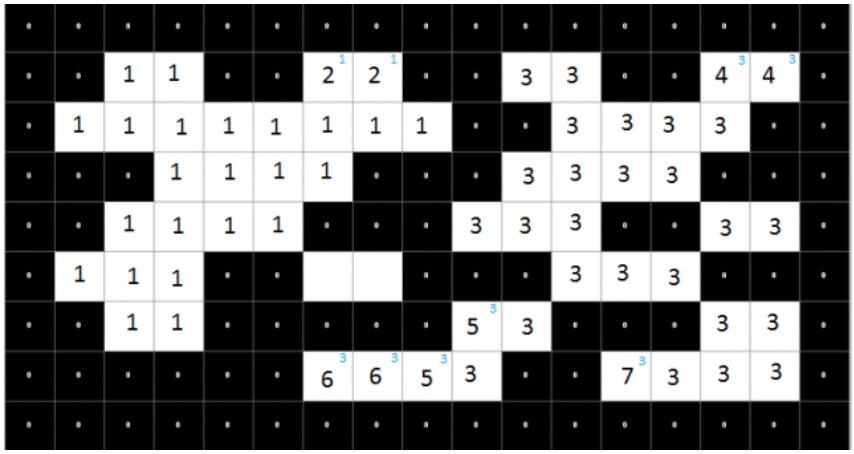
Pod pojmom extrakcia komponent sa skrýva metóda, ktorá rozdelí obraz na jednotlivé objekty, ktoré spolu nie sú spojené. V tomto algoritme sa prejde celý obraz a každý pixel je testovaný na nasledujúce podmienky :

- Ak sa v okolí skúmaneho pixelu nenachádza označený pixel, je skúmanemu pixelu priradená značka.
- Ak sa v okolí skúmaneho pixelu nachádza práve jeden už označený pixel, je skúmanému pixelu priradená značka nájdeneho pixelu.
- Ak sa v okolí skúmaneho pixelu nachádza viac ako jeden označený pixel, vytvorí sa záznam o ekvivalencii.

Záznam o ekvivalenií reprezentuje informáciu o tom, kto je rodičom daného pixelu. Komponentu samotnú predstavuje trieda **Box**. Tá má svoju značku a rodiča. Pri inicializácii je každý *Box* svojim vlastným rodičom. Ak nastane situácia, kedy je nutné vytvoriť záznam o ekvivalencii, priradí sa *Boxu*, ktorý má vyššiu hodnotu značky ako rodič značka druhého boxu. Zároveň má každý *Box* informáciu o všetkých svojich potomkoch. V prípade, že sa zmení rodič *Boxu* ktorý je rodičom, alebo sa zmení rodič niektorého z jeho potomkov, je táto zmena propagovaná ku každému potomkovi a rodičovi. Zmení sa teda rodič všetkým *Boxov*, ktoré majú rovnakého rodiča.

Výsledok tohoto procesu je znázornený na obrázku 24. Záznam o rodičovi je uvedený v pravom hornom rohu.

V ďalšom kroku sa prechádza obraz znova a komponenty sa označia rodičovskými značkami. Výsledkom je teda obraz s rozlíšenými komponentami. Vrámci tohoto procesu sa zisťujú súrad-



Obr. 24: Výsledok prvého kroku extrakcie komponent.

nice štvorca, ktorý ohraničuje oblasť, v ktorej sa komponenta nachádza. Taktiež sa zisťuje v koľkých komponentách sa aktuálna komponenta nachádza.

Vďaka tomuto algoritmu je možné vypĺňať každú komponentu zvlášť, čo vedie k zlepšeniu výsledkov vypĺňania objektu, keďže chyba sa propaguje iba v rámci oblasti, kde sa komponenta nachádza.

Výsledky extrakcie na testovacích datach sú obrazené na obrázku 25 a obrázku 26. Farebné rozlíšenie štvorco značí v koľkých komponentách sa konkrétna komponenta nachádza. Červenou sú zobrazené komponenty prvej úrovne. Modrou sú označené komponenty, ktoré sa nachádzajú v jednej komponente (komponenta druhej úrovne). Napokon zelenou farbou sú ohraničené komponety, ktoré sa nachádzajú na tretej úrovni.

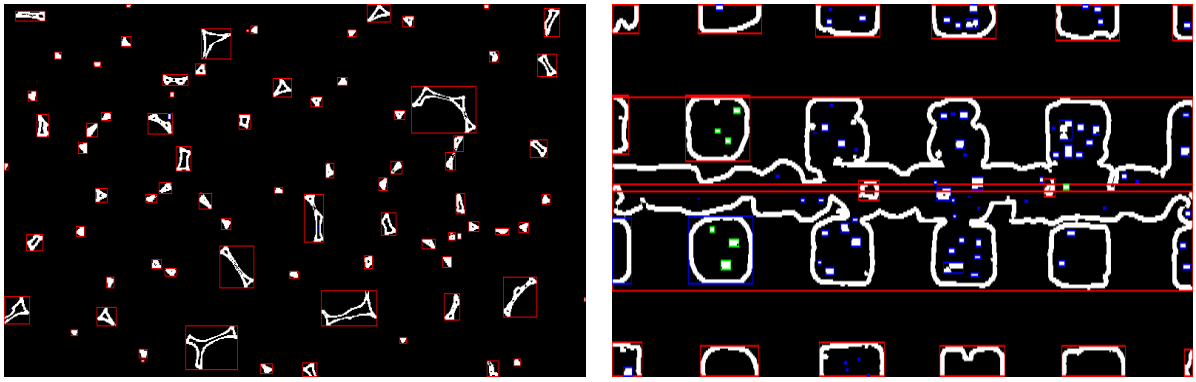
Na obrázku 25b je vidieť problém, s ktorým sa táto metóda potýka. Ako je možné vidieť, stredová komponenta je rozdelená na dve časti. V skutočnosti sa však jedná o jednu spojenú komponentu. Tento problém nastal, pretože stredová komponenta prechádza celou šírkou obrazu a teda komponenta nie je uzavretá hranami. Požiadavka, ktorú teda tento prístup kladie je tá, že komponenty musia byť uzavreté.

6.10 Úprava hrán

Nakoľko predchádzajúci algoritmus kladie ako požiadavku uzavretie hrán, implementoval som do modulu pomocné algoritmy. Jednalo sa o dilatáciu a eróziu. Obe sú jednoduché metódy, ktoré spadajú do morfologických operácií.

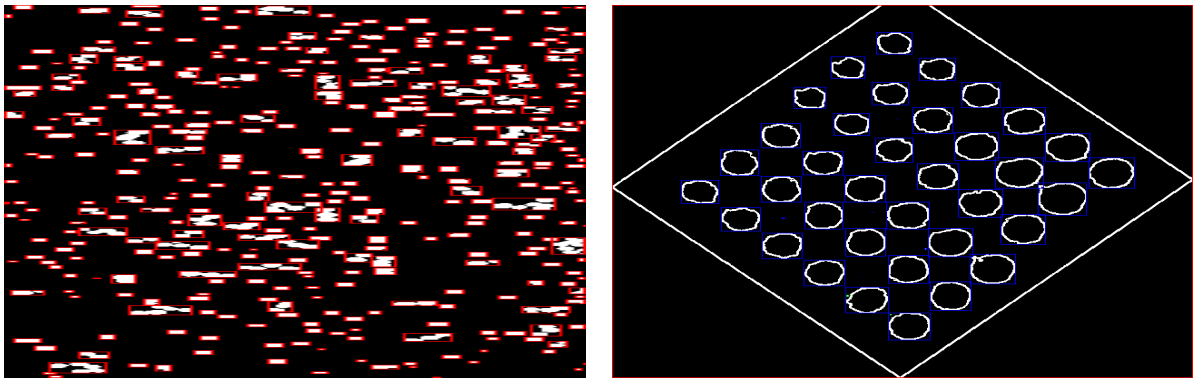
Pomocou dilatácie sa spájajú rozpojené hrany. Nevýhodou je rožšírenie hrany a teda stratíme informáciu o skutočnej pozícii hrany, prípadne zlúčime komponenty, ktoré sú v skutočnosti rozpojené. Algoritmus prejde celý obraz pixel po pixeli a ak je pixel pixelom hrany, sú všetci jeho susedia označený za hranu. Susednosť môžeme definovať 4–okolím alebo 8–okolím.

Hrúbku hrán redukujeme opačnou operáciou a to eróziou. Znova prechádzame obraz bod po bode a ak je jeden zo susedov bodom hrany je odstránený.



(a) (b)

Obr. 25: Výsledky extrakcie komponent.



(a) (b)

Obr. 26: Výsledky extrakcie komponent.

Pre túto prácu bol implementovaný algoritmus **Zhang-Suen**.
Ak skúmame bod **X**, jeho okolie definujeme ako :

$$\begin{bmatrix} H & A & B \\ G & X & C \\ F & E & D \end{bmatrix}$$

ďalej definujme

- $Z(X)$ ako počet zmien farby pixelu v poradí A->B->C->D->E->F->G->H->A.
- $P(X)$ ako počet susedných pixelov objektu.

Algoritmus prejde v dvoch krokoch obraz. Ak v prvom kroku splňuje pixel nasledujúce podmienky je označený ako adept na zmazanie.

1. Pixel je pixelom objektu.
2. $2 \leq P(X) \leq 6$.
3. $Z(X) = 1$.
4. Aspoň jeden z pixelov A, C alebo E je pixel pozadia.
5. Aspoň jeden z pixelov C, E alebo G je pixel pozadia.

V druhom kroku sa testujú nasledujúce podmienky :

1. Pixel je pixelom objektu.
2. $2 \leq P(X) \leq 6$.
3. $Z(X) = 1$.
4. Aspoň jeden z pixelov A, C alebo G je pixel pozadia.
5. Aspoň jeden z pixelov A, E alebo G je pixel pozadia.

Po ich splnení je pixel znova označený za adepta na zmazanie.

Po prejdení sú adepti označení za body pozadia. Tento proces opakujeme pokiaľ neoznačíme žiaden pixel ako adepta na znamenia. Zreťazenie dilatácie a erózie sa nazýva uzavretie. Pri testovaní sa neprejavilo zlepšenie výsledkov, preto tieto metódy nie sú v procese vypĺňania použité. Každopádne sú naďalej poskytované triedou **Segmentation**.

6.11 Plnenie oblasti

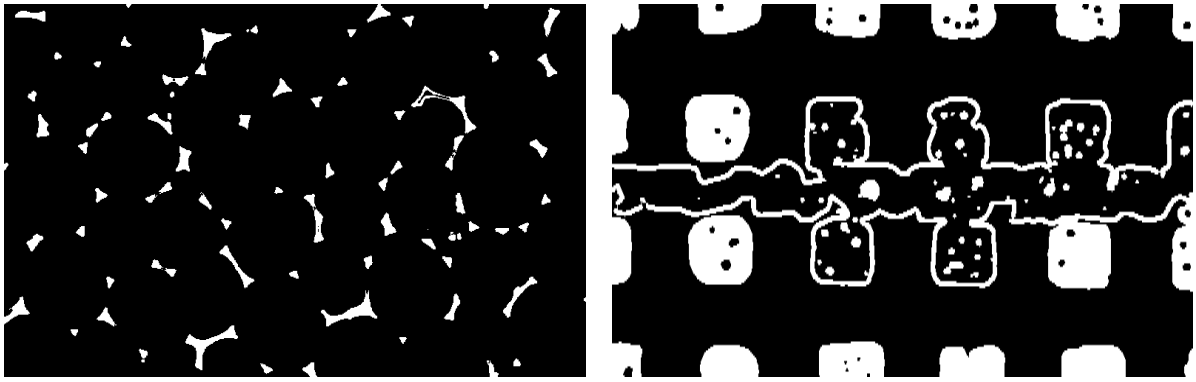
Napriek predchádzajúcim pokusom stále nedosahovala metóda plnenia oblasti na základe počtu priesečníkov uspokojivé výsledky. Preto som použil nasledujúci postup:

Obdĺžnik každej komponenty je zväčšený o jeden pixel vo všetkých smeroch. Táto pridaná oblasť je označená ako *seed*. Každý *seed* je označený za pixel pozadia. Následne sa každý zo susedov *seedu*, ktorý nie je bodom hrany označí za nový *seed*. Tento proces opakujeme pokiaľ na obraze nie je už žiaden *seed*. Týmto spôsobom získame obal objektu. Všetko, čo sa nachádza vnútri tohoto oobjektu je označené za pixel objektu. Tu prichádza na rad informácia o počte komponent, v ktorých sa daná komponenta nachádza. Ak je tento počet nepárny sú vnútorné pixely označené za pixely pozadia, inak sú označené za pixely objektu.

Problém, ktorý nastal pri testovaní je možné vysvetliť na obrázku 26b. Hlavná komponenta nie je z vrchnej časti uzavretá. To predstavuje problém, keďže práve cez túto vrchnú časť sa a vnútro objektu označí ako pixely pozadia, čo vedie k invertovaniu celej vnútornej štruktúry komponenty. Na vyriešenie tohoto problému bol použitý práve prvotný prístup plnenia a to metóda počtu priesečníkov s priamkou. Tá sa aplikuje iba na komponenty, ktoré sa nachádzajú

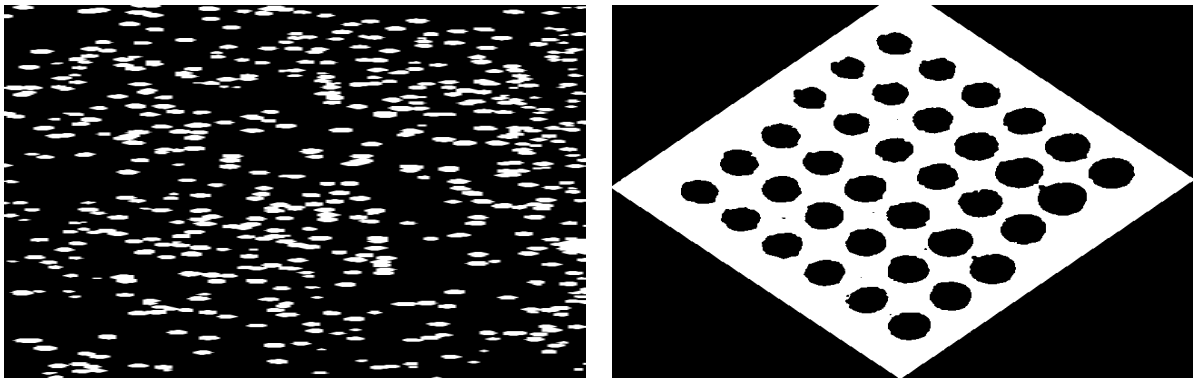
na okraji obrazu a iba na hranu ktorá patrí k okraju. Rovnaký prístup bol použitý aj na obrázok 25b.

Výsledky celého algoritmu sú na obrázkoch 27 a 27.



(a) (b)

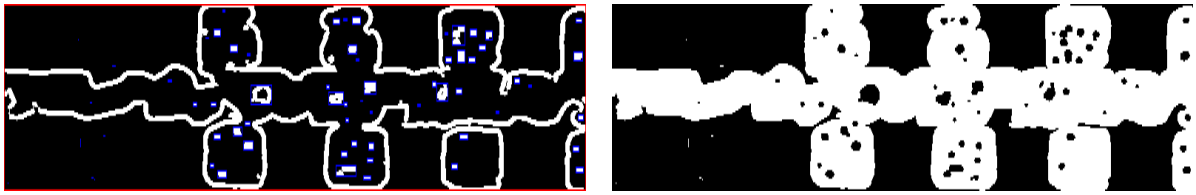
Obr. 27: Výsledky plnenia komponent.



(a) (b)

Obr. 28: Výsledky plnenia komponent.

Na obrázku 29 je ukázané ako vyzerá vypĺňanie stredovej časti obrázku 25b ak ručne prepojíme tieto dve komponenty.



(a) Hľadanie komponent. (b) Vypĺňanie komponenty.

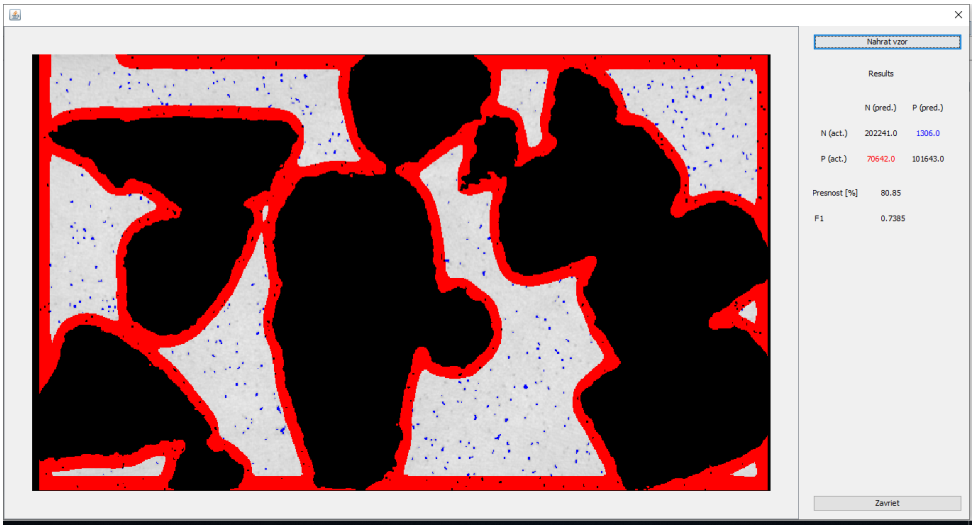
Obr. 29: Výsledky algoritmu pre upravenú komponentu.

6.12 Úprava Watershed algoritmu

Pri tejto metóde sa mi nepodarilo dosiahnúť uspokojivé výsledky ani pomocou procesu vypĺňania oblasti. Výsledky obsahovali hrany aj v tmavých oblastiach a tvorili sa veľké záplavové regióny. Vychádza to z podstaty algoritmu, kde sa obraz delí na záplavové oblasti, čo znamená, že by bolo nutné dodatočne rozlíšiť, či sa jedná o oblasť objektu alebo pozadia. Navyše vznikali hrádze aj medzi oblasťami rovnakého druhu. Preto som sa rozhodol upraviť tento algoritmus pre účely tejto práce. V prvom rade sa obraz delí iba na dve záplavové oblasti – objekt a pozadie. Uživateľ definuje tri hodnoty : minimum, maximum, minimum objektu. Ďalšou úpravou je vytvorenie reliéfu. Ak má bod nižšiu úroveň jasu ako minimum, pridá sa do povodia pozadia. Ak je úroveň jasu väčšia ako hodnota maximum je jeho hodnota invertovaná na hodnotu $255 - úroveň$. Ak je výsledok tejto inverzie menší ako hodnota minimum objektu je tento pixel pridaný do povodia objektu. Po inicializácii sa postupuje štandadným spôsobom. Keďže existujú iba dve povodia, vznikajú hrádze iba medzi nimi.

7 Testovanie

V tejto časti je predstavená metodika testovania a dosiahnuté výsledky. Pre testovacie účely bola vytvorená vzorka 14 obrazov tak, aby zahŕňali čo najväčšiu škálu prípadov obsiahnutých v celej sade, ktorá mi bola poskytnutá. Testovacia vzorka bola najskôr upravená pomocou upraveného binárneho prahovania. Úprava spočíva v tom, že pixely z hodnotu väčšou ako prah ostávajú bez zmeny. Následne boli v snímkoch ručne označené pixely pozadia, ktoré neboli spracované prahovaním. Pre možnosti testovania bola implementovaná metóda na porovnávanie dvoch obrazov. Ukážka testovacieho okna je zobrazená na obrázku 30. Za negatívne instancie sú považované pixely s hodnotou jasu 0, za pozitívne všetky ostatné. Toto okno môže používať uživateľ, ak má vzor, ako má vyzerať výsledok segmentácie.



Obr. 30: Testovacie okno.

Keďže by ručné testovanie zdĺhave a mám k dispozíci metriku na určenie úspešnosti segmentácie voči vzoru (F1 skóre), je možné chápať túto úlohu ako optimalizačnú. Z tohoto dôvodu som implementoval algoritmus SOMA.

7.1 SOMA

SOMA (SamoOrganizujúci sa Migračný Algoritmus) je optimalizačný genetický algoritmus, ktorý využíva ku svojej funkcií prácu s populáciou riešení. Nevytvárajú sa však nový jedinci, ale dochádza k ich migráci v stavovom priestore. Samoorganizácia vzniká vzájomným ovlpyvňovaním jedincov populácie pri pohybe. Kým pri iných algoritmoch predstavujú parametre jedinca jeho genóm a dochádza ku kríženiu a mutáciam, pri algoritme SOMA predstavujú tieto parametre súradnice v hyperploche riešení.

Na začiatku náhodne vygenerujeme určitý počet jedincov. Následne vyhodnotíme ich vhodnosť a určíme najlepšieho jedinca. Toho označíme za *lídra*. Následne sa každý jedinec okrem

lídra pokúsi nájsť vhodnejšie riešenie. Toto hľadanie prebieha v smere k lídrovi po určtých skokoch. Cesta k lídrovi má určitú dĺžku. Po tejto ceste robí jedinec skoky určitej veľkosti. Na každej pozícii skoku sa vyhodnotí jeho vhodnosť. Po vykonaní všetkých skokov zaujme jedinec pozíciu, v ktorej dosiahol najväčšiu vhodnosť. Tieto presuny však nemusia nutne prechádzať po priamke. Pri každej migrácii je vygenerovaný vektor, ktorý určuje, ktoré parametre sa nebudú počas skoku meniť. V priestore je môžné si tento proces predstaviť ako odbočky od priamky. To nám zaručí väčšiu diverzitu.

Po presune všetkých jedincov končí cyklus hľadania. Jeden takýto cyklus sa nazýva *epocha*.

7.2 F1 skóre

V predchádzajúcej časti bolo uvedené, že vhodnosť jedinca určíme pomocou F1 skóre. Toto skóre vychádza z takzvanej matice zámien (*confusion matrix*). Táto matica obsahuje informáciu o počte správnych a chybných klasifikáciach. Príklad takejto matice je uvedený v tabuľke 1

Tabuľka 1: Príklad matice zámien.

	Predikcia (-)	**Predikcia (+)**
(-)	A	B
(+)	C	D

A predstavuje počet správnych predikcií, že daná instancia je negatívna.

B predstavuje počet nesprávnych predikcií, že daná instancia je pozitívna.

C predstavuje počet nesprávnych predikcií, že daná instancia je negatívna.

D predstavuje počet správnych predikcií, že daná instancia je pozitívna.

Následne môžeme pomocou vzorcov 42 až 44 zaviesť nasledovné:

$$ACCURACY = \frac{(A+D)}{(A+B+C+D)} \tag{42}$$

$$PRECISION = \frac{D}{(D+B)} \tag{43}$$

$$RECALL = \frac{D}{(D+C)} \tag{44}$$

Výpočet *ACCURACY* predstavuje najjednoduchšiu metódu pre porovnanie úspešnosti. Jej veľkou nevýhodou je fakt, že nezohľadnuje prípady, kedy sú počty pozitívnych a negatívnych instancií nevyvážené. V takomto prípade je táto hodnota často zavádzajúca.

PRECISION je definovaná ako pomer správnych predikcií pozitívnych instancií voči všetkým, ktoré boli predikované ako pozitívne. *RECALL* predstavuje pomer správnych predikcií pozitívnych instancií voči všetkým pozitívnym instanciam

Následne môžeme definovať F1 skóre pomocou vzorca 45

$$F1 = 2\frac{(PRECISION * RECALL)}{(PRECISION + RECALL)} \tag{45}$$

$$\tag{46}$$

Výsledné skóre môže nadobúdať hodnoty od 0 po 1, kde 1 predstavuje najlepší výsledok a 0 najhorší.

7.3 Výsledky

Testovanie prebiehalo tak, že pre každý obraz bola testovaná každá kombinácia filtračnej metódy (vrátane vypnutej filtrácie) a segmentačnej metódy. Následne bola použitá SOMA pre optimalizáciu parametrov metód a optimalizáciu parametrov histogramových operácii. Nastavenie algoritmu SOMA je uvedené v tabuľke 2.

Tabuľka 2: Konfigurácia algoritmu SOMA.

	Počet epoch	Počet jedincov	Veľkosť cesty	Veľkosť skoku
Hodnota	2	3	1	0.4

Pôvodne boli počiatočné parametre odlišné, testovanie bolo však kôli tomu časovo príliš náročné. Z toho dôvodu boli použité uvedené hodnoty.

V tabuľkách 3 a 4 sú uvedené výsledné najlepšie hodnoty F1 skóre, ktoré daná segmentačná metóda dosiahla počas testovania. V tabuľke 5 je uvedené priemerne, minimálne a maximálne skóre, ako aj rozdiel minimálneho a maximálne skóre, kroré daná segmentačná metóda dosiahla na všetkých obrazoch.

Z tabuľky 5 sa dá usúdiť, že najoptimálnejšou metódou je K-Means, keďže dosiahol najväčší priemer. Taktiež je vidieť že si táto metóda viedla veľmi dobre na všetkých obrazoch testovacej množiny, keďže rozdiel minimálneho a maximálneho skóre je iba 0.093. Ďalšou silnou metódou je Watershed. Tá mala síce horší priemer ale dosahovala stabilnejšie výsledky naprieč celou sadou.

Za mierne sklamanie považujem Otsuho prahovanie, keďže je táto metóda podobná binárnemu prahovaniu, čakal som podstatne vyšší priemer. Síce dosiahla ideálne F1 skóre, dosiahla zároveň aj jedno z najnižších skóre. Naopak vyššie skóre ako som očakával dosiahli segmentačné metódy založené na prvej deriváci (s výnimkou Robertsovho filtra). Naopak kompasové varianty dosiahli výrazne nižšie skóre napriek podobnému princípu fungovania.

Sklamaním sú určite metódy založené na druhej deriváci. Na druhú stranu sú tieto metódy zložitejšie na nastavenie, čo mohlo viesť k tomu, že optimalizačný algoritmus nepreskúmal dostatočne veľkú polochu.

Tabuľka 3: Výsledné hodnoty F1 skóre segmentačných metód.

Obraz	0	1	2	3	4	5	6	7
Binárne prahovanie	0.994	0.989	0.476	0.899	0.749	0.325	0.940	0.984
Otsuho prahovanie	0.910	0.895	0.079	0.100	0.621	0.263	0.629	0.505
Robertsov filter	0.794	0.851	0.675	0.635	0.891	0.263	0.158	0.175
Sobelov filter	0.966	0.967	0.867	0.855	0.991	0.978	0.499	0.946
Prewittovej filter	0.977	0.969	0.859	0.587	0.989	0.325	0.499	0.976
Scharr filter	0.976	0.969	0.861	0.760	0.986	0.980	0.981	0.977
Kompas.f. Kirsch	0.916	0.506	0.126	0.367	0.598	0.262	0.228	0.209
Kompas.f. Sobel	0.983	0.517	0.341	0.277	0.588	0.284	0.340	0.141
Kompas.f. Roninson	0.938	0.732	0.135	0.605	0.392	0.559	0.145	0.410
Frei Chen	0.845	0.884	0.626	0.678	0.716	0.501	0.269	0.365
Laplacian	0.519	0.187	0.244	0.306	0.195	0.119	0.701	0.467
Nevati Babu	0.928	0.697	0.597	0.619	0.121	0.181	0.467	0.225
LoG	0.508	0.072	0.219	0.169	0.060	0.057	0.276	0.634
DoG	0.517	0.130	0.184	0.424	0.327	0.068	0.191	0.365
Canny	0.918	0.781	0.586	0.685	0.399	0.271	0.989	0.135
SUSAN	0.788	0.549	0.087	0.524	0.593	0.241	0.499	0.358
K–Means	0.989	0.982	0.908	0.904	0.996	0.903	0.967	0.947
Watershed	0.990	0.992	0.902	0.897	0.950	0.891	0.925	0.895

Tabuľka 4: Výsledné hodnoty F1 skóre segmentačných metód.

Obraz	8	9	10	11	12
Binárne prahovanie	0.817	0.986	0.988	0.963	0.989
Otsuho prahovanie	1.000	0.871	0.968	0.954	0.982
Robertsov filter	0.457	0.525	0.221	0.309	0.968
Sobelov filter	0.869	0.986	0.978	0.979	0.983
Prewittovej filter	0.955	0.982	0.939	0.972	0.983
Scharr filter	0.731	0.985	0.972	0.976	0.983
Kompas.f. Kirsch	0.320	0.615	0.124	0.788	0.825
Kompas.f. Sobel	0.531	0.449	0.384	0.149	0.984
Kompas.f. Roninson	0.245	0.523	0.132	0.838	0.931
Frei Chen	0.567	0.266	0.716	0.515	0.969
Laplacian	0.325	0.263	0.092	0.602	0.557
Nevati Babu	0.318	0.967	0.403	0.577	0.864
LoG	0.397	0.131	0.067	0.431	0.272
DoG	0.186	0.147	0.468	0.142	0.817
Canny	0.597	0.367	0.923	0.658	0.815
SUSAN	0.579	0.817	0.938	0.848	0.825
K–Means	0.959	0.989	0.968	0.963	0.996
Watershed	0.898	0.945	0.982	0.981	0.992

Tabuľka 5: Hodnoty jednotlivých segmentačných metód.

Obraz	Priemer	Minimum	Maximum	Rozdiel
Binárne prahovanie	0.854	0.325	0.994	0.661
Otsuho prahovanie	0.675	0.079	1.000	0.921
Robertsov filter	0.533	0.158	0.968	0.810
Sobelov filter	0.913	0.500	0.991	0.491
Prewittovej filter	0.847	0.325	0.990	0.665
Scharr filter	0.934	0.731	0.990	0.258
Kompas.f. Kirsch	0.453	0.124	0.916	0.792
Kompas.f. Sobel	0.459	0.141	0.984	0.843
Kompas.f. Roninson	0.506	0.132	0.938	0.806
Frei Chen	0.609	0.266	0.969	0.704
Laplacian	0.352	0.092	0.701	0.610
Nevati Babu	0.536	0.121	0.967	0.846
LoG	0.253	0.057	0.634	0.576
DoG	0.305	0.068	0.817	0.749
Canny	0.623	0.135	0.989	0.854
SUSAN	0.588	0.087	0.938	0.852
K–Means	0.959	0.903	0.996	0.093
Watershed	0.941	0.891	0.992	0.010

8 Záver

Cieľom tejto práce bolo vytvorenie modulu pre systém FOTOM NG, schopného vykonávať segmentáciu kovov, za účelom analýzy ich štrukturálnych vlastností. Vzhľadom na potrebu pridania tohoto modulu do systému FOTOM NG je modul implementovaný v jazyku Java s dôrazom na modularitu jednotlivých častí. To zároveň umožnuje jednoduché ďalšie rožšírenie tohoto modulu.

Modul obsahuje samostantý modul pre aplikovanie filtrov na obraz, pre prácu s histogramom ako aj samostantý modul pre segmentáciu obrazu. Taktiež obsahuje modul pre získavanie obrazu z DICOM súborov.

V úvodnej časti sú nahrané dáta a vykonané histogramové operácie. Tie môžu pomôcť odstrániť určité artefakty, ktoré by mohli viesť ku chybnej segmentácii. Následne je vykonaná filtrácia za účelom odstránenia šumu z obrazu. Celkovo bolo implementovaných šesť filtračných metód. Všetky z nich sú plne parametrizovatelné. Po predspracovaní obrazu má uživateľ možnosť výberu zo všetkých segmentačných metód, ktoré boli spomenuté v teoretickej časti tejto práce. Tieto metódy sú taktiež parametrizovatelné. Vzhľadom na snahu o čo najväčší uživatelský zážitok je vo veľkej miere využitý paralelizmus pre urýchlenie spracovania.

Počas implementácie som riešil najmä problém s návrhom postupu ako získať z mapy hrán objekt. Vďaka vyriešeniu tohoto problému som mohol vyskúšať použitie metód založených na vytváraní mapy hrán. Taktiež bola implementovaná modifikovaná Watershed metóda pre účely tejto práce. O týchto skutočnostiach pojednáva kapitola, ktorá sa venuje implementáci.

Pre účely testovania bola implementovaná optimalizačná metóda SOMA, pri ktorej sa určovala vhodnosť jedinca pomocou F1 skóre. Výsledky testovania a jednotlivých metód sú zhrnuté v predposlednej kapitole tejto práce. Z nich vyplíva, že dvoma najlepšími metódami sú metódy K-Means a Watershed. Taktiež sa ukázalo, že pomerne vyššie skóre dosahujú aj základné metódy segmentácie založené na prvej deriváci. Naopak metódy založené na druhej deriváci rovanko ako metóda SUSAN a Cannyho hranový detektor sa ukázali ako nevhodné pre účely segmentácie kovov.

Celkovo za najlepšiu segmentačnú metódu považujem K-Means segmentáciu, nakoľko pri metóde Watershed sa v určitých prípadoch prejavuje vyššia časová náročnosť. Navyše považujem nastavenie K-Means segmentácie za jednoduchšie.

V budúcej práci by som sa rád venoval možnému zlepšeniu prevodu mapy hrán na objekt. Za zaujímavú by som považoval možnosť automatickej voľby segmentačnej metódy ako aj jej parametrov na základe povahy snímku.

Výstup tohoto modulu je použitý pre ďalšiu analýzu vlastnosti kovových štruktúr, a vytváranie ich 3D modelov.

Pre systém bola vypracovaná uživateľská príručka a programátorská dokumentácia, ktoré sú súčasťou prílohy.

Literatúra

[1] *KROUPOVÁ, I., RADKOVSKÝ, F., LICHÝ, P.* **Slévárenské technologie výroby kovový pěn s nepravidelným uspořádaním vnitřních dutin**

[2] *CHEN, L., PAPANDREOU, G., KOKKIONS, I., MURPHY, K., YUILLE, A. L.* **DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs**
IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, vol. 40, no.4, s. 834-848, 2018.

[3] *CHOUHAB, S.S, KAUL, A., SINGH, U.P.* **Image Segmentation Using Computational Intelligence Techniques: Review**
Archives of Computational Methods in Engineering, 2018.
https://doi.org/10.1007/s11831-018-9257-4

[4] *DHABACHANDRA, Nameirakpam, CHANU, Yambem Jina.* **A New Image Segmentation Method Using Clustering and Region Merging Techniques**
Springer Singapore, 2019
ISBN 978-981-13-1819-1.

[5] *VEDANARAYANAN, V., NANDHITHA, N.M.* **Advanced image segmentation techniques for accurate isolation of abnormality to enhance breast cancer detection in digital mammographs**
esearch Article - Biomedical Research, vol. 28, 2017.
http://www.biomedres.info/biomedical-research/advanced-image-segmentation-techniques-html

[6] *HORÁK, Karel, KALOVÁ, Ilona, PETYOVSKÝ, Petr, RICHTER, Miroslav.* **Počítačové vidění**
Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2008
http://www.uamt.feec.vutbr.cz/vision/TEACHING/MPOV/Pocitacove_videni_S.pdf

[7] *DOBEŠ, Michal.* **Zpracování obrazu a algoritmy v C#**
Praha: BEN - technická literatura, 2008
ISBN 978-80-7300-233-6.

[8] *ŠRÁMEK, Jaromír, RÁČEK, Ondřej, SEDLÁŘ, Martin, MORNSTEIN, Vojtěch.* **Získávání a analýza obrazové informace**
Masarykova univerzita v Brně, Lékařská fakulta – Biofyzikální ústav, 2011
https://www.med.muni.cz/biofyz/Image/ucebnice.pdf

[9] *PÁTEK, Pavel.* **Aplikace MPEG-7 deskriptorů při analýze 3D biomedicínských obrazových dat**
Masaryková univerzita, Fakulta informatiky, Brno, 2013
`https://theses.cz/id/0niqqf`

[10] *HONG, Lin, WAN, Yifei, Jain, Anil.* **Fingerprint Image Enhancement: Algorithm and Performance Evaluation**
IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, vol. 20, no.8, s. 777-789, 1998.

[11] *MARR, D. HILDERTH, E. :* **Theory of edge detection**
Proc. Royal Soc. Lond., volume B 207, str.: 187-217, 1980.

[12] *SMITH, S.M. BRADY, J.M. :* **SUSAN - a new approach to low level image processing**
International Journal of Computer Vision, 23(l):45-78, 1997.

[13] *SBEUCHER, S.I. :* **Image Segmentation and Mathematical Morphology.**
`http://www.cmm.mines-paristech.fr/~beucher/wtshed.html`

[14] *BEUCHER, S., LANTUÉJOUL, CH :* **Use of watersheds in contour detection. In International workshop on image processing, real-time edge and motion detection**
`http://cmm.ensmp.fr/~beucher/publi/watershed.pdf`

[15] *ROERDINK, J.B.T.M, MEIJSTER, A :* **The Watershed Transform**
`http://www.cs.rug.nl/~roe/publications/parwshed.pdf`

[16] **Java Platform SE 7, Class ForkJoinPool**
Oracle, 2014
`http://docs.oracle.com/javase/7/docs/api/java/util/concurrent/ForkJoinPool.html`

Zoznam príloh

Prílohy k tejto práci sú uložené na priloženom CD. Jedná sa o :

1. Zdrojové kódy systému
2. Uživateľská príručka pre modul
3. Programátorská príručka pre modul.